# SIMULATION OF A ROBOT VISION SYSTEM BASED ON TWO - DIMENSIONAL IMAGE ANALYSIS

*by*

A. VASUDEVAN

DEPARTMENT OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY, KANPUR

MARCH, 1986

# SIMULATION OF A ROBOT VISION SYSTEM BASED ON TWO - DIMENSIONAL IMAGE ANALYSIS

*A Thesis Submitted*
in Partial Fulfilment of the Requirements
for the Degree of

MASTER OF TECHNOLOGY

*by*

A. VASUDEVAN

*to the*

DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY, KANPUR
MARCH, 1986

Th
629.8920724
V449 S

EE-1986-M-VAS-SIM

CERTIFICATE

This is to certify that this thesis entitled, 'SIMULATION OF A ROBOT VISION SYSTEM BASED ON TWO-DIMENSIONAL IMAGE ANALYSIS', by A. Vasudevan has been carried out under my supervision and that it has not been submitted elsewhere for the award of a degree.

[illegible]

March, 1986

( R.N. Biswas )
Professor
Department of Electrical Engineering
Indian Institute of Technology
KANPUR

## ACKNOWLEDGEMENT

I would first like to sincerely thank Dr. R.N. Biswas for his guidance during the project. The long hours of discussions with him have been valuable.

I am grateful to Mr. U. Guha for helping me in making the hardware.

Further, I wish to thank Mr. Prem Malhotra, Research Engineer, for helping me in the Microcomputer Development System Laboratory.

Finally, I express my appreciation to Mr. J.S. Rawat for the neat typing of the thesis.

- A. Vasudevan

CONTENTS

LIST OF FIGURES

# CHAPTER 1
# INTRODUCTION

The potential applications of sensor-based robotics are many. Although other sensing schemes like force, tactile and active or passive compliance have important roles to play, vision appears to offer the richest source of sensory information for robotic manipulation in the widest range of environments.

Vision systems for robot guidance are basically of two forms:

i) Identification of parts and position determination for assembly and parts handling. It is based on real time computer interpretation of images involving recognition of the part, computation of its centre and orientation and the passing of this information in real time to a robot.

ii) The second class of robotic vision guidance task is visual servo-control. The vision system is used as a position sensor to control the movement of a mechanism in a closed-loop. Seam tracking for arc welding is one example.

In this thesis, a vision system for assembly and parts handling is considered. The functional block diagram of a robotic vision system for such an application is given in Fig. 1.1.

In the acquisition phase, it is important to obtain an image which makes the recognition task as easy as possible. In most cases, the boundary of an object is adequate for recognition. This is because all objects are exact templates of the reference object. There is no question of distortion as in a character recognition application.

In this project, the two-dimensional image of an object has been used for analysis. The recognition scheme is based on the boundary of the object. The boundary is detected by simple threshold comparison. Since most industrial parts have a polygonal shape, geometric features are employed resulting in considerable simplicity in computation. The number of vertices of the polygon, the sequence of relative angles and the sequence of the lengths of the sides are used as features of the object. The location and orientation of the object are also determined.

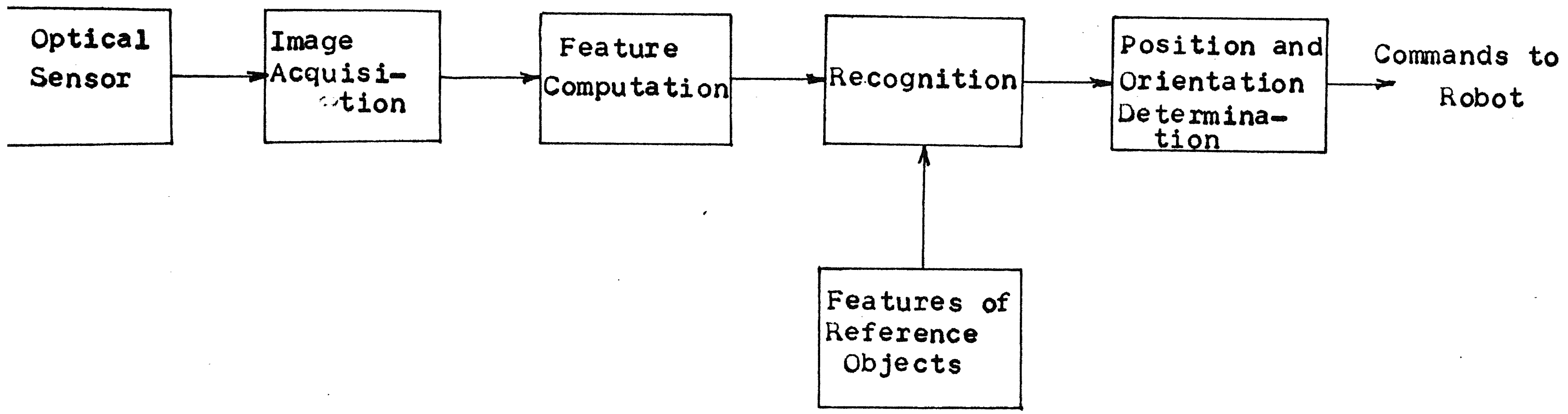

Fig. 1.1: A Typical Robotic Vision System

Fig. 1.2 gives the overall system block diagram. The generation of image data has been simulated in software. For simulation purposes, the camera is assumed to be fixed to the robot arm. The movement of the gripper is simulated by displaying the modified images acquired by the camera as the gripper moves. A byte-mapped graphics display card has been used for this purpose.

The system can handle all objects with two-dimensional images of convex polygonal shape. Objects with holes are also accepted but the information about the holes is not used for recognition purpose .

Chapter 2 gives an overview of robot vision schemes. Chapter 3 explains the overall system - the image acquisition hardware, feature selection and extraction and the recognition scheme used. The simulation of the system is explained in Chapter 4. Software details are provided in Chapter 5. Conclusion and results form the last chapter.

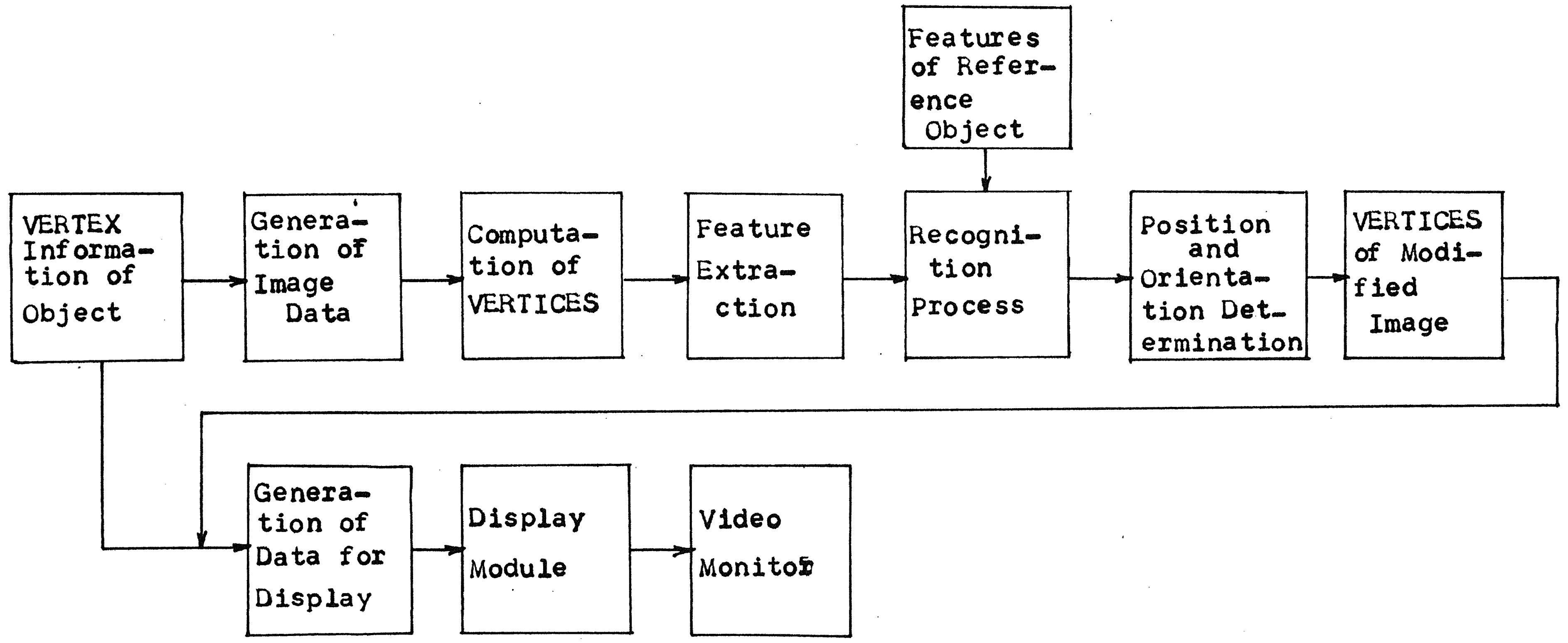
VERTEX Informa- tion of Object
Genera- tion of Image Data
Computa- tion of VERTICES
Feature Extra- ction
Features of Refer- ence Object
Recogni- tion Process
Position and Orienta- tion Det- ermination
VERTICES of Modi- fied Image
Genera- tion of Data for Display
Display Module
Video Monitor

Fig. 1.2: Block Diagram of the Overall System

# CHAPTER 2

# AN OVERVIEW OF ROBOT VISION SYSTEMS

The use of robots has become a well-established and reliable way to increase the flexibility and degree of automation in industrial manufacturing. The high flexibility is provided by the use of sensory control. One class of intelligent sensors is the vision system. A review of robot vision systems is done in this chapter.

## 2.1 REQUIREMENTS OF A ROBOT VISION SYSTEM:

The basic function of a robot vision system is to identify an object and to determine its location. Since the object can be located anywhere within a predefined area, it must be possible to identify the object irrespective of its location. Finding the object's location involves determining its position and its orientation.

The next requirement is that the image processing should be done in real time. Real-time processing means that the processing time should not affect the overall cycle time of the system. The processing of gray-scale images at high resolution often provides impressive results but this is inevitably achieved at the expense of processor architecture and processing time. So a fast and simple image processing algorithm is needed.

The system must be able to work in an industrial environment. It must therefore be insensitive to normal variations in ambient light and function reliably in a factory illumination. Finally, the system should be reliable and cost-effective.

## 2.2 SENSORY DEVICES FOR VISION:

Some of the sensory devices available are: photo-detectors - either in a linear array or in an area array and TV camera.

The main criteria to be applied in selecting the sensory devices are - cost, ruggedness and size. Since the only satisfactory location of the sensor is near the end-effector, the sensing devices must be small enough to be integrated close to the gripper jaw.

Solid-state arrays are used only in a limited way due to their high cost. Photo-detector arrays of modest resolution are obsolete because of the limited application range. TV cameras are the most widely used vision sensors because of their low cost and good resolution; their size disadvantage is also fast disappearing.

## 2.3 OBJECT RECOGNITION SCHEMES:

In this section some recognition schemes are described and their relative merits and demerits are evaluated.

### 2.3.1 Circles Model [1]:

A useful parameter in the identification of the shape of the objects is the angular position of the maximum and minimum radius vectors. A radius vector is defined as a line from the centre of area (centroid) of an object to a point on the edge of the object. But for some shapes the radius vector may change only slowly about the maximum and minimum values as the outline is followed. In such situations, the positions of these vectors determined will be unreliable.

A more fruitful approach is to specify a value of radius, during the learning phase, for which clearly defined vector points can be determined. In effect a circle of given radius is superimposed on the object, centred at the centre of area. The intersection of this circle with the outline are defined as feature points.

In order to uniquely define the orientation of a component or to differentiate it from another similar part, it may be necessary to specify more than one circle. The intersection points found when the outline is traced round will then form an ordered list of radius - angle pairs. In the example shown in Fig. 2.1, two radii are used.

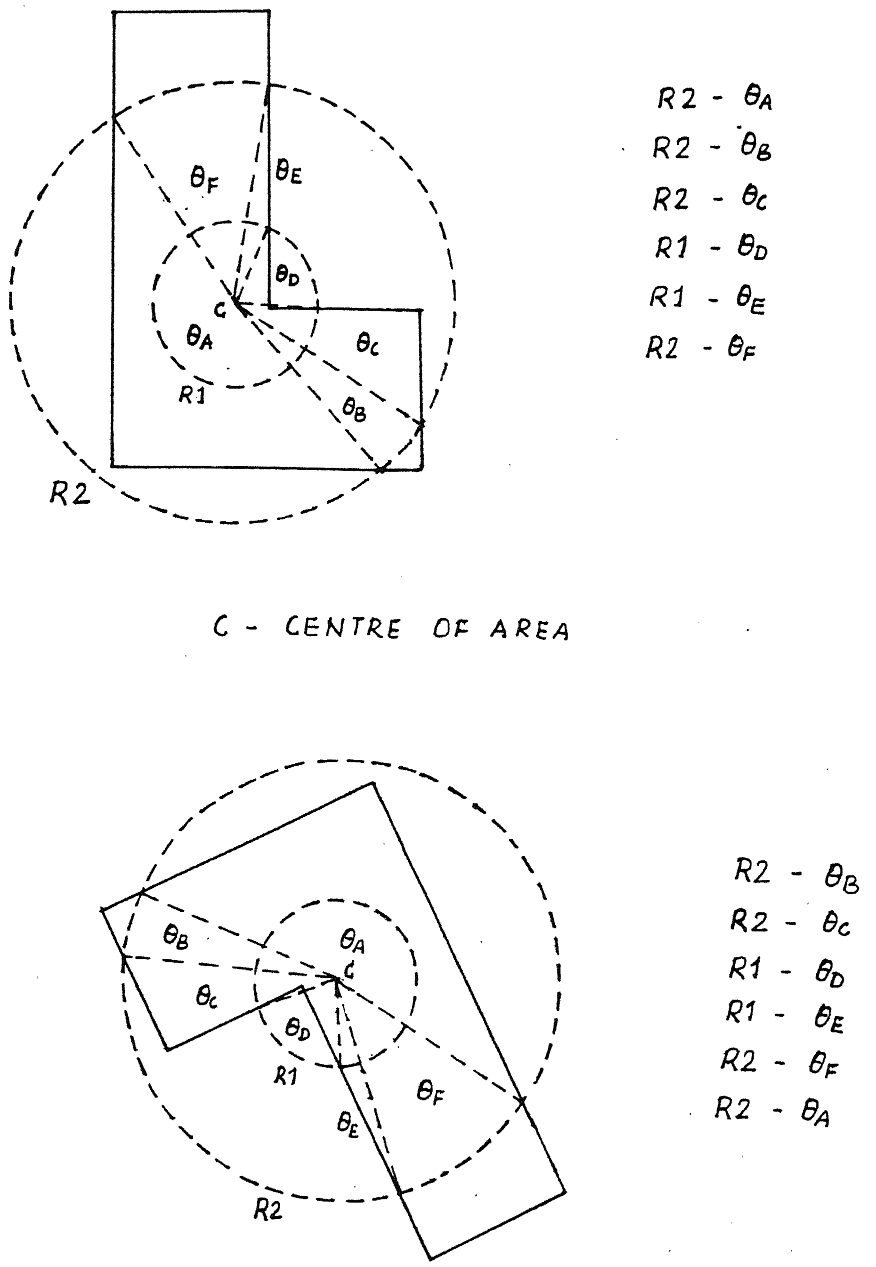

Fig. 2.1: Radius-angle list for CIRCLE model

To compare the two images, the two lists are first checked to ensure that they have the same number of feature points. Then they are compared for a match both in radius and in relative angle entries. If no match is found, one list is rotated and the comparison is repeated. When the correlation between the two lists is established, the difference between the absolute angular positions of each pair of corresponding intersection points can be calculated. The mean of these values is a measure of the orientation of one object relative to the other.

The circles model cannot be used for an object that has been scaled down in size with respect to the reference object. The distance between the object and the camera should remain constant as in the case of systems using static overhead cameras.

In the learning process, the radii of the circles have to be specified. The radii have to be selected on the basis of certain criteria. For instance, intersection points should not vanish or new points appear for small changes in radius. So human interaction is required.

## 2.3.2 Fourier Descriptors Method [2]:

The problem of recognition is one of extracting a finite set of numerical features from a closed curve. Fourier descriptors can be used as features that will tend to separate the shapes of different classes relative to intra-class dispersion.

A starting point on the boundary is selected and a function is defined which measures angular deviation of the curve as a function of arc length. After appropriate normalization this periodic function is expanded in a Fourier series and the coefficients of a truncated expansion are used as shape features called Fourier descriptors.

Assume C to be a clockwise-oriented simple closed curve with parametric representation $(x(l), y(l))$ where $l$ is the arc length and $0 \leq l \leq L$. Denote the angular direction of C at point $l$ by the function $a(l)$. The cumulative angular function $f(l)$ is defined as the net amount of angular deviation between the starting point and point $l$ as shown in Fig. 2.2; i.e.,

$$f(l) = a(l) - a(o) - 2n\pi \quad \text{for } nL \leq l \leq (n+1)L \quad \text{where } n = 0,1,2,3,\ldots$$

Fig. 2.3(b) shows $f(l)$ for a shape depicted in Fig. 2.3(a).

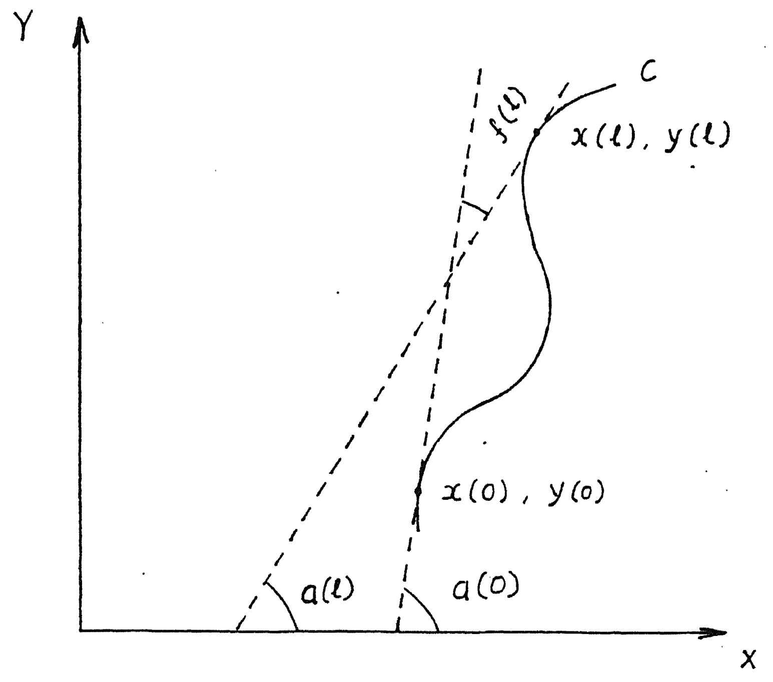

Fig. 2.2: Representation of a plane curve with tangential direction a(l) and cumulative angular function f(l)

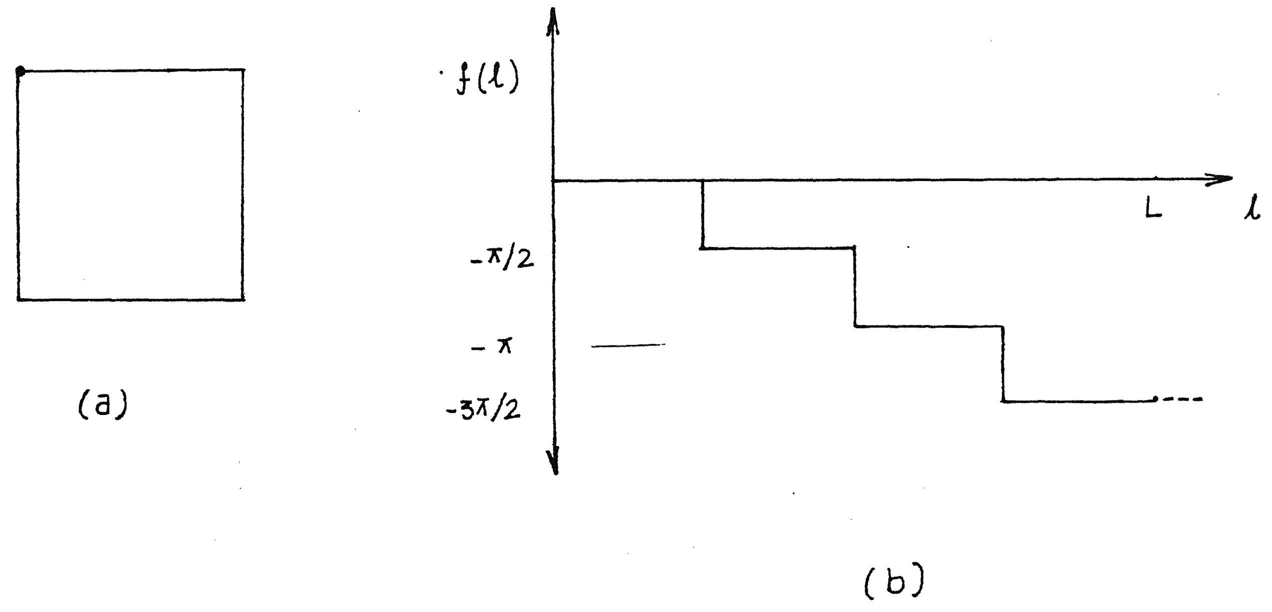

Fig. 2.3: (a) Plane closed curve in the form of a square and (b) its cumulative angular function f(l)

It can be seen that $f(o) = 0$ and $f(L) = -2\pi$ because all simple closed curves with clockwise orientation have a net angular deviation of $-2\pi$.

The domain of definition $o \leq l \leq L$ of $f(l)$ simply contains absolute size information. So it has to be normalised to the interval 0 to $2\pi$ which is standard for periodic functions.

For this purpose, a normalised function $f^*(t)$ is defined with t as the normalised variable,

$$t = \frac{2\pi l}{L} \quad \text{and}$$

$$f^*(t) = f(l) + t$$

$$= f\left(\frac{Lt}{2\pi}\right) + t : 0 \leq t \leq 2\pi$$

$f^*(t)$ is defined such that

$$f^*(o) = f^*(2\pi) = 0.$$

Expanding $f^*(t)$ as a Fourier series

$$f^*(t) = \mu_o + \sum_{k=1}^{\infty} (A_k \cos(kt - p_k))$$

where,

$(A_k, p_k)$, $k = 1,2,\ldots,$ are theFourier descriptors of the curve C.

$f^*(t) = o$ for a circle. The function $f^*(t)$ for different curves are shown in Fig. 2.4. The function $f^*(t)$ measures the way in which the shape in question differs from a circular shape. It can be seen from Fig. 2.4, that the function $f^*(t)$ also reflects the rotational symmetry. Thus all simple plane closed curves with the same starting point are mapped into the class of periodic functions over the domain 0 to $2\pi$ in such a way that all curves of identical shape and starting point generate the same function $f^*(t)$.

$f^*(t)$ is invariant under translation, rotation and scaling. In this method, position and size is factored out in the values of some parameters.

For C and C' (same curves) with different starting points $(X_o, Y_o)$ and $(X'_o, Y'_o)$, the Fourier descriptors $(A_k, p_k)$ and $(A'_k, p'_k)$ of C and C' satisfy

$$A_k = A'_k$$

$$p'_k = p_k + k \Delta p$$

$$\text{and} \quad \Delta p = -2\pi \frac{\Delta l}{L}$$

where $\Delta l$ is the clockwise arc length along the curve from $C_o$ to $C'_o$. It is a measure of orientation.

But the disadvantage of the Fourier descriptors method is that $f^*(t)$ for polygonal curves contains discontinuities and

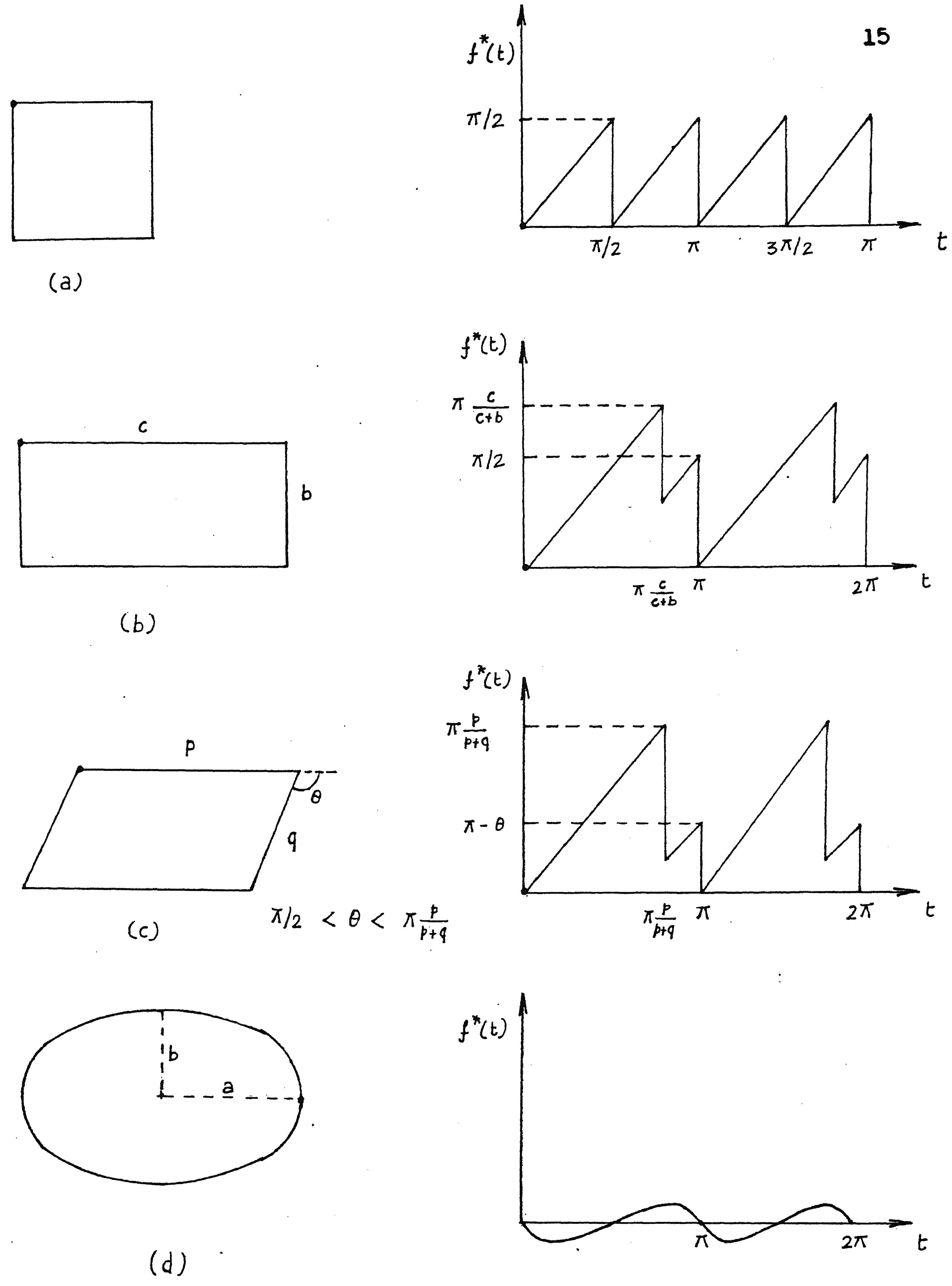

Fig. 2.4: Normalised cumulative angular function for (a) a square (b) a rectangle, (c) a parallelogram and (d) an ellipse.

so $A_k$ will decrease rather slowly as k increases. Identification of shapes will be fast if the number of coefficients are less. For objects having sharp corners, as most industrial parts are, the number of Fourier descriptors that have to be used in identification are large. So the method of Fourier descriptors is not well suited for such objects.

Thus we have seen some schemes for object recognition and position determination. In the next chapter, a new scheme is proposed. It is similar to these schemes in the sense that it also utilises the information about the contour of the object. The contour is coded to reduce storage space and to facilitate handling of the data. In contrast to the schemes discussed, this scheme can handle only objects having convex polygonal shape and curved objects are handled by using straight line approximations.

# CHAPTER 3

# ROBOT VISION SYSTEM CONFIGURATION

The objective of the thesis is to develop a vision system for use in sensor-based robots. The task of the vision sensor in this application involves object recognition and position calculation. As suggested in the last chapter, a TV camera is used as the sensor in the system. The object is assumed to have a good contrast with respect to the background and a threshold is used to trace the contour of the object. All further evaluations like image analysis, part recognition and position determination are based on the acquired contour data. The following sections present the hardware configuration and the methods of feature extraction, pattern recognition and orientation determination used in this system.

## 3.1 IMAGE RESOLUTION DETERMINATION:

The object's contour is detected by comparing the video signal with a threshold. The output of the comparator for one particular scan line of an object is shown in Fig. 3.1.

To obtain the contour information, the time intervals $T_X$ and $T_D$ shown have to be converted to proportional digital

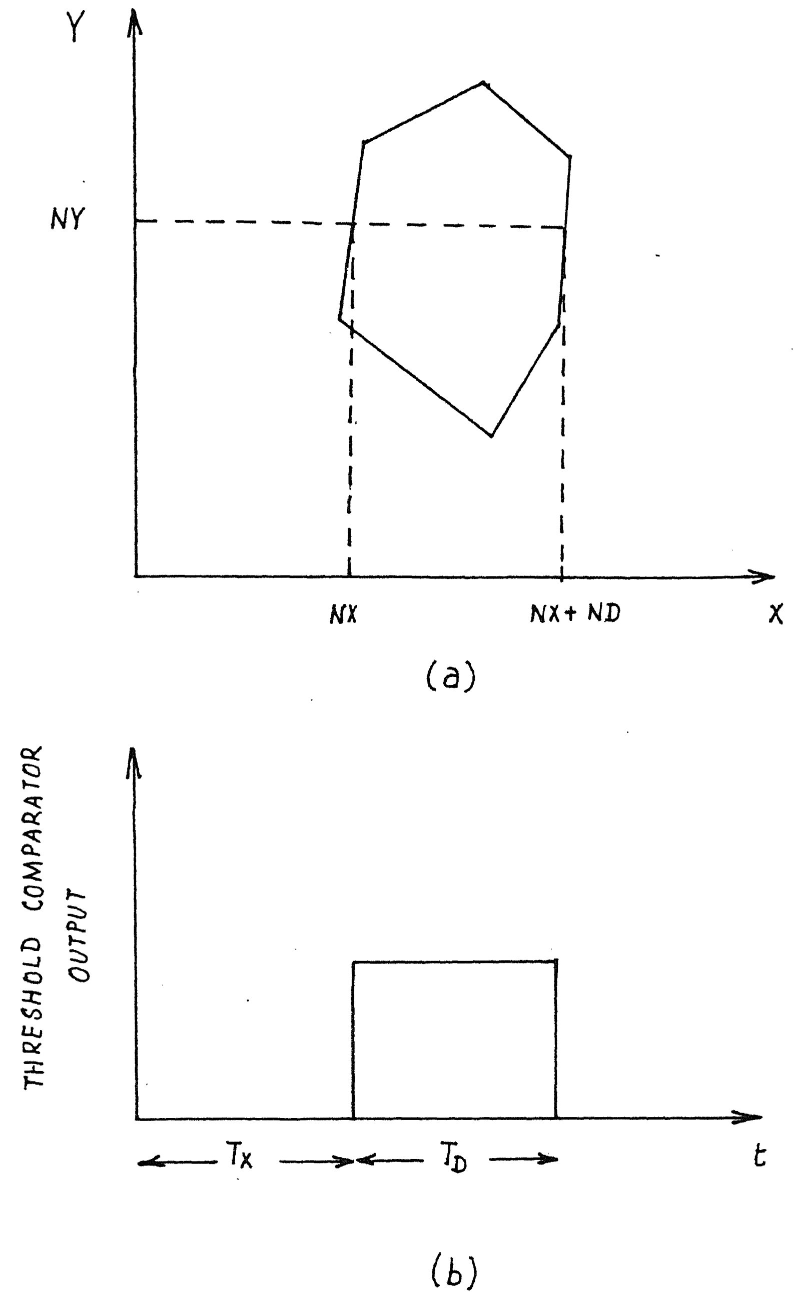

Fig. 3.1 a) 2-D image of an object and
b) Threshold comparator output for the scan line NY.

quantities using counters. The clock signal to these counters determines the resolution in terms of the points per line that can be distinguished. A resolution of about 500 pixels (dots) per line is considered reasonable for the purpose of recognition. For a line scan time of 60-65 microseconds, the visible time would be about 50 microseconds giving a time duration of 0.1 microseconds per pixel. So a dot clock frequency of 10 MHz has been chosen in this system.

## 3.2 IMAGE ACQUISITION SYSTEM:

The hardware block diagram of the image acquisition system is shown in Fig. 3.2.

At the beginning of each scan line, a counter CX is reset to zero. The dot clock runs the counter CX. When the video signal from the camera crosses the threshold, the counter CX is stopped and another counter CD is started. The count value NX in CX gives a measure of the position of the left side of the object with respect to an arbitrary zero column taken as the Y-axis.

The second counter CD receives the same dot clock. It counts until the end of that line. Each negative going edge of the comparator output loads the count value in a register. The count value in the register (ND) gives a measure of the width of the object for that scan line.

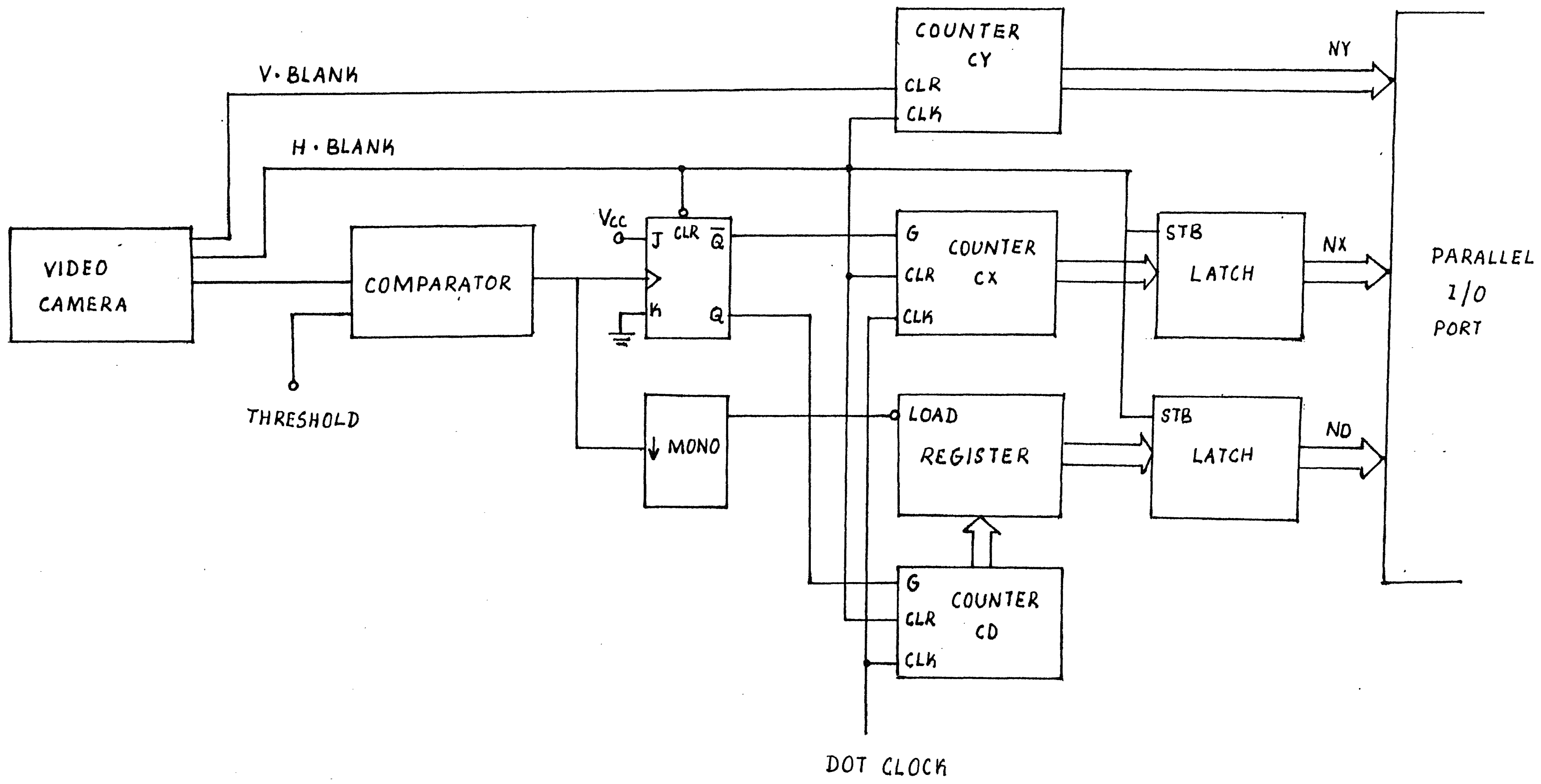

Fig. 3.2: Image Acquisition System

A third counter CY counting the horizontal sync. pulses is reset to zero at the beginning of each frame. This counter counts the lines and gives the Y-co-ordinate NY of the counter points.

For a TV frame consisting of N lines, we thus have 2N data words, each line having corresponding values of NX and ND. This set of data, corresponding to one video frame, will henceforth be referred to as Video Data.

## 3.3 SYSTEM CONSTRAINTS:

As the hardware handles only two transitions in one scan line, no re-entrancy is allowed, that is only objects having the shape of convex polygons are accepted. The system does not accept objects having curved contours.

The object is assumed to be isolated within the field of view of the camera. The object is either stationary or slowly moving. The lag between the measured position of the object and the true position of the object should be small enough to cause no error when gripping of the object takes place.

The video camera is fixed rigidly to the robot gripper. The main advantage in such an arrangement is that there will be correlation between the camera's field of view and the co-ordinate system of the robot. Overhead static cameras require a tedious procedure to establish the camera to robot reference co-ordinates. This is often a time-consuming operation. With the 'eye in hand' approach, only a tool-offset transformation is required. This is a one-time measurement of the relationship of the gripper to the camera axis.

## 3.4 FEATURE SELECTION:

The features have to be selected on the basis of the system requirements. In order to recognize one particular object among a class of objects, the descriptors used must be position and orientation invariant because a component may be presented to the assembly machine in any orientation and any position in the field of view. Moreover, the features must be amenable to fast computation.

In this system, geometric features are employed. For polygon-shaped objects, the number of sides of the polygon is useful in classifying the object.

A position and orientation invariant parameter is the relative angle. The relative angle denotes the angle of deviation of a line segment from a previous line segment of the polygon.

It is possible for two different objects to have the same sequence of relative angles. So the lengths of the line segments are also included in the feature vector.

In order to find the location of the object, it is usual practice to compute the centroid of the object and position the gripper at that point. In this system, an apriori knowledge of the gripping sides is used to determine the 'centre' of the object. For this purpose, the 'centre' is taken to be the midpoint of the line connecting the midpoints of the gripping segments themselves. (Fig. 3.3). The length of this line passing through the 'centre' of the object gives a measure of the width of the gripper needed. Hence, the gripping segments are also elements of the feature vector.

## 3.5 EXTRACTION OF FEATURES OF THE OBJECT:

The Video Data obtained from the acquisition system is used to determine the features of the object. As a first step, the vertices of the polygonal image have to be determined.

For the purpose of finding the vertices, a line segment is considered to be made up of several overlapping 'small segments'. For each contour point under consideration, these 'small segments' are formed by joining that point and, say, the next three contour points in clockwise order. For example, in Fig. 3.4, contour points, 1,2,3 and 4 will form

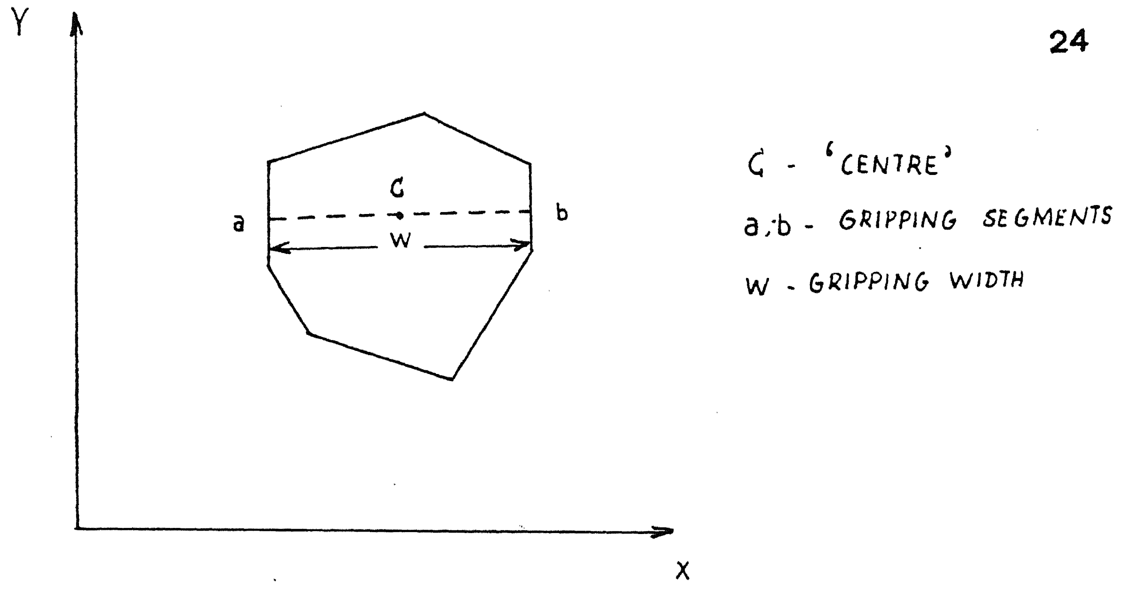

Fig. 3.3: 'Centre' of the object

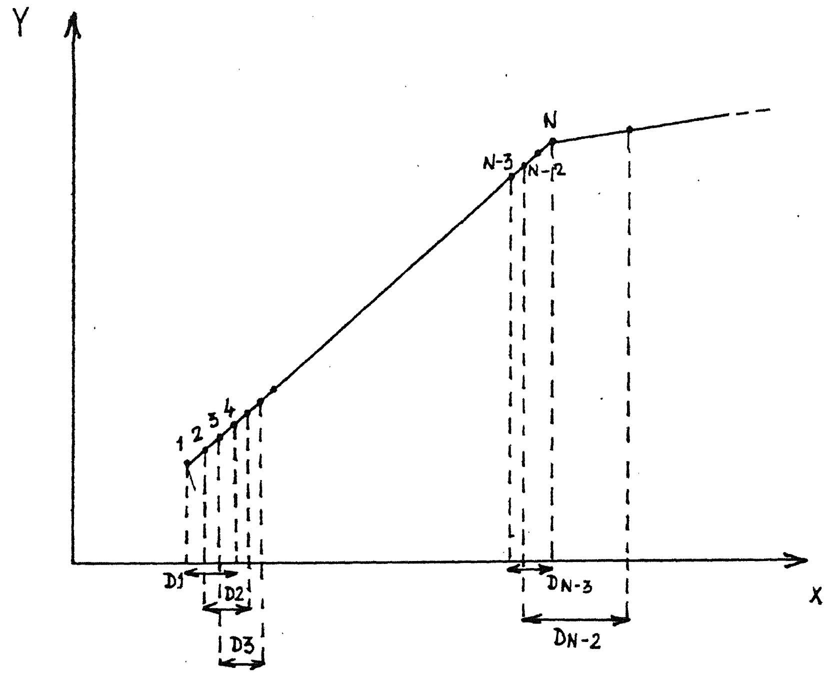

Fig. 3.4: Determination of a vertex

one 'small segment', points 2,3,4 and 5 another and so on. The projections of these 'small segments' on the X-axis are equal for all 'small segments' within that line segment of the polygon. These projections along the X-axis are denoted by $D_1$, $D_2$,...$D_N$. It can be seen from Fig. 3.4 that

$$D_1 = D_2 = D_3 = \dots = D_{N-3} \neq D_{N-2}$$

As the 'small segment' starting from the contour point (N-2) does not satisfy the specified condition, one of the contour points forming it is a vertex. By examining the points within that segment for a change of slope or comparing each with their adjacent points, a vertex can be found.

The number of contour points in a 'small segment' is varied for each side of the polygon to get better results.

Using the vertices of the object, other features are computed. Consider a polygon of n sides as shown in Fig. 3.5. Let $V_i$, i=1,2,...n, denote the vertices of the polygon in clockwise order. The ith line segment of the polygon is the line joining the ith and i+1th vertex. The vertices of the polygon are denoted by $(X_i, Y_i)$, i=1,2,...n.

The absolute angle $a_i$ of the ith line segment is obtained by considering that ith line segment as a vector

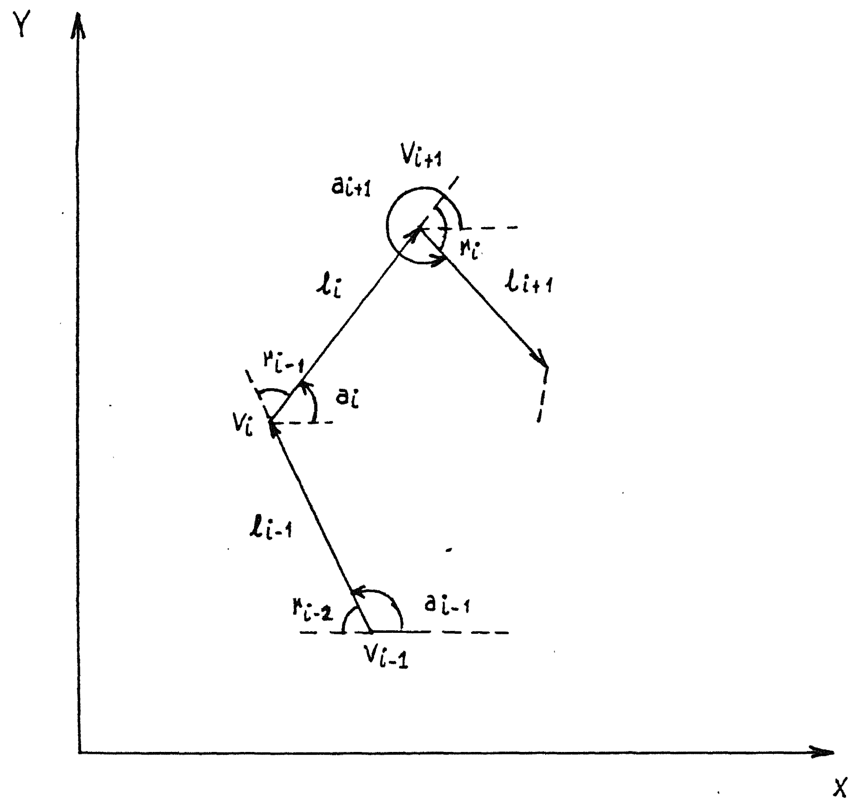

Fig. 3.5: Features of the object

pointing towards the i+1th vertex and finding the direction of this vector. It is measured anti-clockwise with respect to the horizontal (Fig. 3.5).

The absolute angles can be computed as follows:

$$\text{Let} \quad b_i = \sin^{-1}\left( \frac{|Y_i - Y_{i+1}|}{\sqrt{(Y_i - Y_{i+1})^2 + (X_i - X_{i+1})^2}} \right)$$

Then,

$$a_i = b_i, \text{ for } Y_{i+1} \geq Y_i,\ X_{i+1} \geq X_i$$

$$a_i = \pi - b_i, \text{ for } Y_{i+1} \geq Y_i,\ X_{i+1} < X_i$$

$$a_i = \pi + b_i, \text{ for } Y_{i+1} < Y_i,\ X_{i+1} < X_i$$

$$a_i = 2\pi - b_i, \text{ for } Y_{i+1} < Y_i,\ X_{i+1} \geq X_i \qquad (3.1)$$

The set of absolute angles $a_i$, $i=1,2,\ldots n$, is used in computing the relative angles of the polygon.

The relative angle $r_i$ between ith and i+1th line segments is given by

$$r_i = a_i - a_{i+1} \qquad (3.2)$$

The length of the line segments are denoted by $l_i$, $i=1,2,\ldots n$. The computation of $l_i$ is straightforward.

$$l_i = \sqrt{(X_i - X_{i+1})^2 + (Y_i - Y_{i+1})^2} \qquad (3.3)$$

## 3.6 IDENTIFICATION OF THE OBJECT AND DETERMINATION OF THE ORIENTATION:

A three-stage hierarchical classification scheme is used for object recognition. It is first checked if objects with similar number of vertices are stored. Then the object shapes are matched on the basis of the sequence of relative angles.

Let the sequence of relative angles of the reference object be

$$R = r_1, r_2, \ldots, r_n.$$

(n denotes the number of sides of the polygon) and the sequence of the relative angles for the unknown object be

$$S = s_1, s_2, \ldots s_n.$$

Then the similarity of shapes can be measured by the Euclidean distance between these two sets:

$$d(j) = \sqrt{\sum_{i=1}^{n} (r_i - s_{i+j})^2} \qquad (3.4)$$

$$j = 0, 1, \ldots, n-1$$

A shift of the sequence S by the parameter j implies a rotation of the unknown object with respect to the stored object. The best fit of the curves will occur at the minimum of the function d(j). Let k be the value of j for minimum d(j). Then d(k) gives the extent of similarity of the object shapes and is used for object recognition. k determines the orientation as given by the angle of rotation between the recognised object and its stored reference.

It is possible for two different objects to have the same sequence of relative angles. So the length of the sides of the polygon are also compared. Using k the corresponding sides of thetwo polygons can be found. For matching to be independent of scaling, the ratio of the length of the side to the perimeter of the polygon is used. If they match then the object is said to be recognized.

# CHAPTER 4

# SIMULATION OF THE VISION SYSTEM

The vision system has been discussed in the previous chapter. The simulation of this system is described in this chapter.

## 4.1 SIMULATION OF THE VISUAL SENSOR SYSTEM:

The system has been developed in the Intellec Series III Microcomputer Development System. Due to the difficulty in interfacing the proposed hardware with the Development System, the generation of Video Data is simulated.

The software module 'OBTAINDATA' generates the Video Data. The vertices of the polygonal image of the object are provided by the user. The video data is of the same form as the one obtained from the proposed hardware system. It consists of two words for each scan line.

As a first step towards simulation, the resolution of the imaging device is fixed. A resolution of 512 pixels per line and 256 lines per frame is used in this system. The whole software can be adapted for any other camera resolution by changing the values of two parameters - 'NPIXEL' denoting the number of pixels per line and 'NLINES' denoting the number of lines per frame.

Next, the reference axes are chosen. The bottom most scan line corresponds to the X-axis and the leftmost column serves as the Y-axis. (Fig. 4.1).

The co-ordinates of the vertices $(X_i, Y_i)$, i=1,2,...n should lie within the specified range

$$0 \leq X_i \leq \text{NPIXEL}$$
$$0 \leq Y_i \leq \text{NLINES}$$

The vertices are given in the order in which they occur if the contour is traced clockwise starting from any arbitrary vertex. The video data generation is thus simulated by specifying the the vertices instead of placing the object and acquiring its image.

## 4.2 DISPLAY SYSTEM:

The image of the object as seen by the camera can be viewed on a video monitor. If a camera is used, then the video signal from the camera will drive the monitor directly. But in this system only the coded Video Data is available. So display hardware is necessary to transform the digital image into a video signal for display.

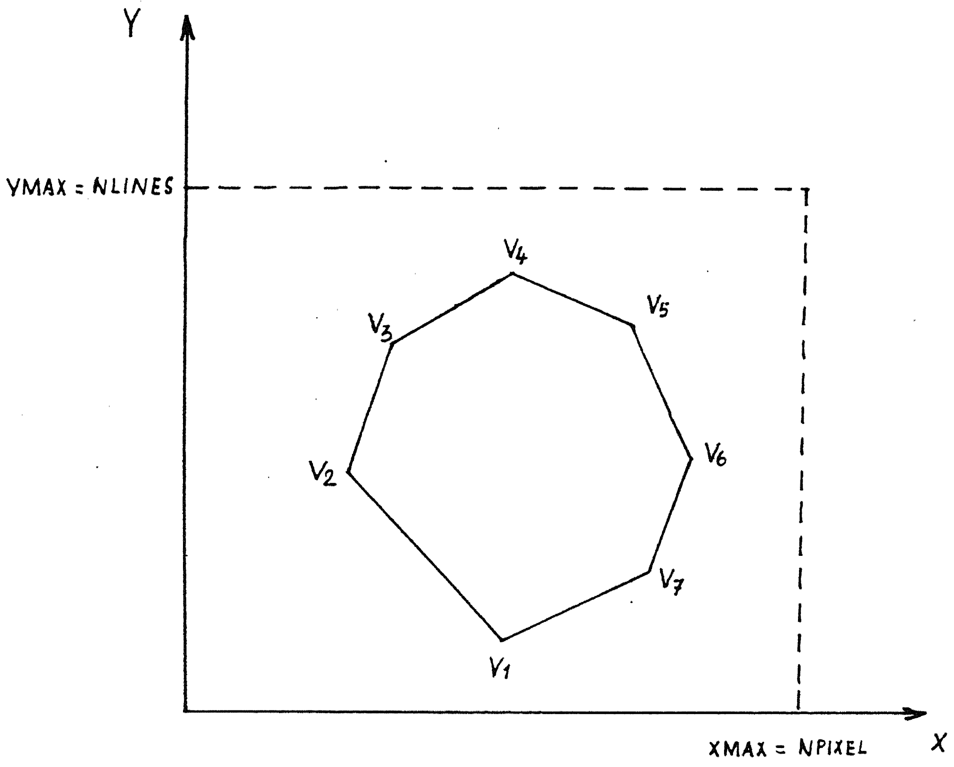

Fig. 4.1: Field of view of the camera

The minimum hardware requirements include a 256x2 word memory (word length depends on the horizontal resolution), counters and comparators. The block diagram of such a system is shown in Fig. 4.2.

But the drawback in this case is that any future upgradation of software will call for a new display card. Hence a more general purpose hardware is designed. In this display system, each pixel is represented by one bit of memory. An RS-232C asynchronous serial link is used to download the video data from the Microcomputer Development System to the display system. The display card has been made compatible with the 8085A processor based workstation developed at I.I.T., Kanpur. The hardware details of the display section is treated in a separate section.

As the camera is fixed with the gripper, a movement of the gripper results in the movement of the camera also and this is reflected in the image displayed on the video monitor. Hence a movement of the gripper along the co-ordinate axes by a specific amount causes a translation of the image by the same amount. The rotation of the gripper about a point is also seen on the image. So the gripper action can be simulated by making an equivalent transformation on the image.

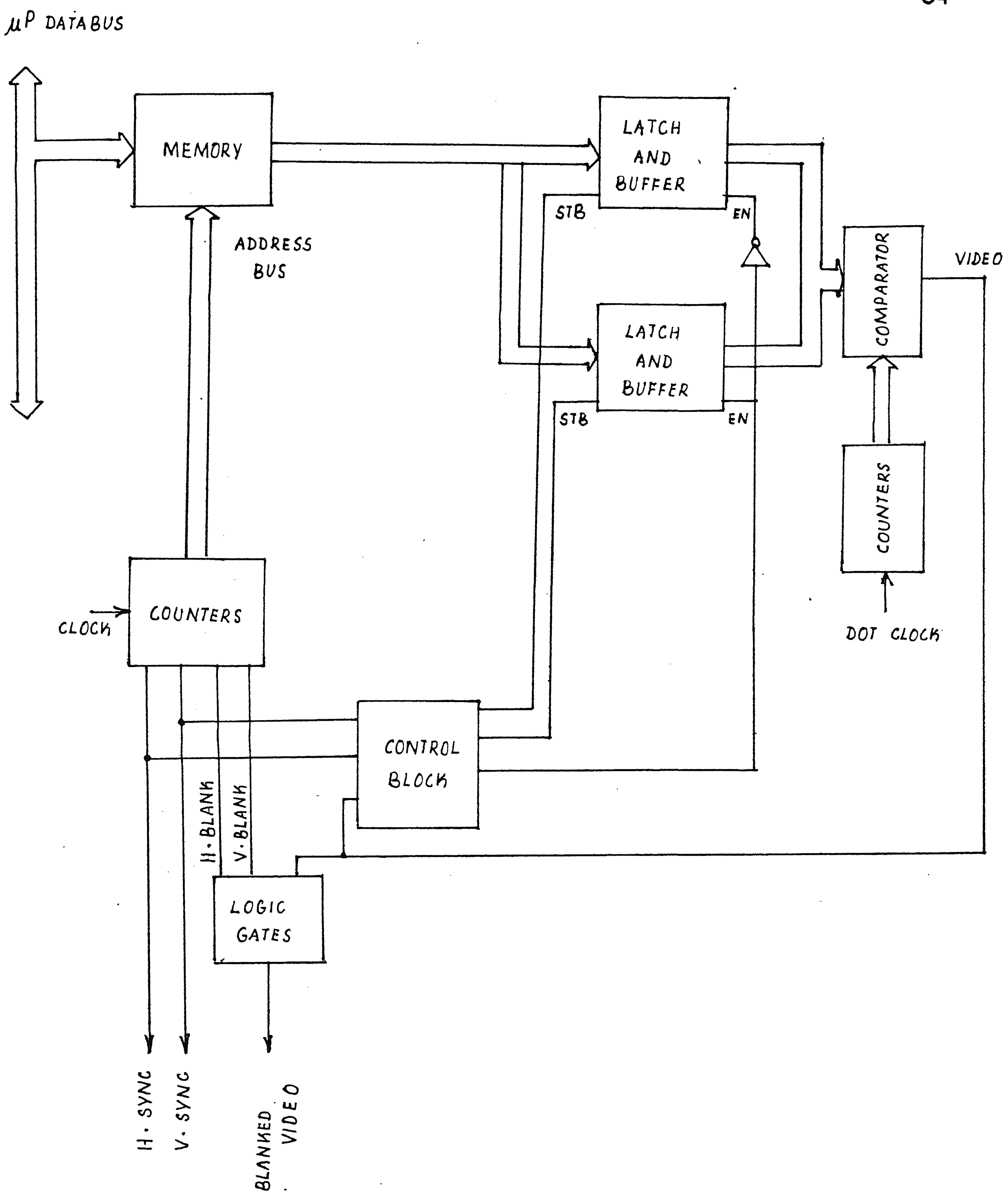

Fig. 4.2: Block diagram of the display system

## 4.3 VISION SYSTEM SOFTWARE:

The vision system software has been written in PASCAL-86. It has been implemented in a modular fashion. The different software routines are listed below:

### 4.3.1 Learning Process Routine: (LEARN)

In the learning process, the features of any new object specified are computed and stored. This routine uses the non-main modules listed below:

Video Data Generation Module (OBTAINDATA):

This is used to generate the Video Data as explained in Sec. 4.1.

Feature Extraction Modules (FINDVERTEX, PARAMETERS):

Using the Video data, the vertices are found and the geometric features are computed.

### 4.3.2 Recognition Process Routine (RECOGNITION):

This module is used to identify the unknown object. The programme returns a message as to whether the object is, or is not, recognised.

A three-stage classification scheme is used for object recognition. The details of the recognition scheme are explained in the next chapter.

The input to this routine is the Video Data of the unknown object. This is generated by the non-main module (ROTAT). The vertices of the polygonal image, the rotation angle and the scaling factor are provided by the user. The image is rotated about the point which is the centre of the field of view of the camera.

## 4.3.3 Robot Control Process Routine:

When the object is recognised, its location has to be determined. For this purpose, the 'centre' of the object $(X_c, Y_c)$ is computed (Fig. 4.3). This centre is then moved to the centre of the field of view given by (NP2, NL2) where,

$$\begin{aligned} NP2 &= NPIXEL/2 \\ NL2 &= NLINES/2 \end{aligned} \tag{4.1}$$

The following translations along the co-ordinate axes are made:

$$\begin{aligned} T_x &= NP2 - X_c \\ T_y &= NL2 - Y_c \end{aligned} \tag{4.2}$$

A point (x,y) in the original image is transformed to (x',y') where

$$\begin{aligned} x' &= x + T_x \\ y' &= y + T_y \end{aligned} \tag{4.3}$$

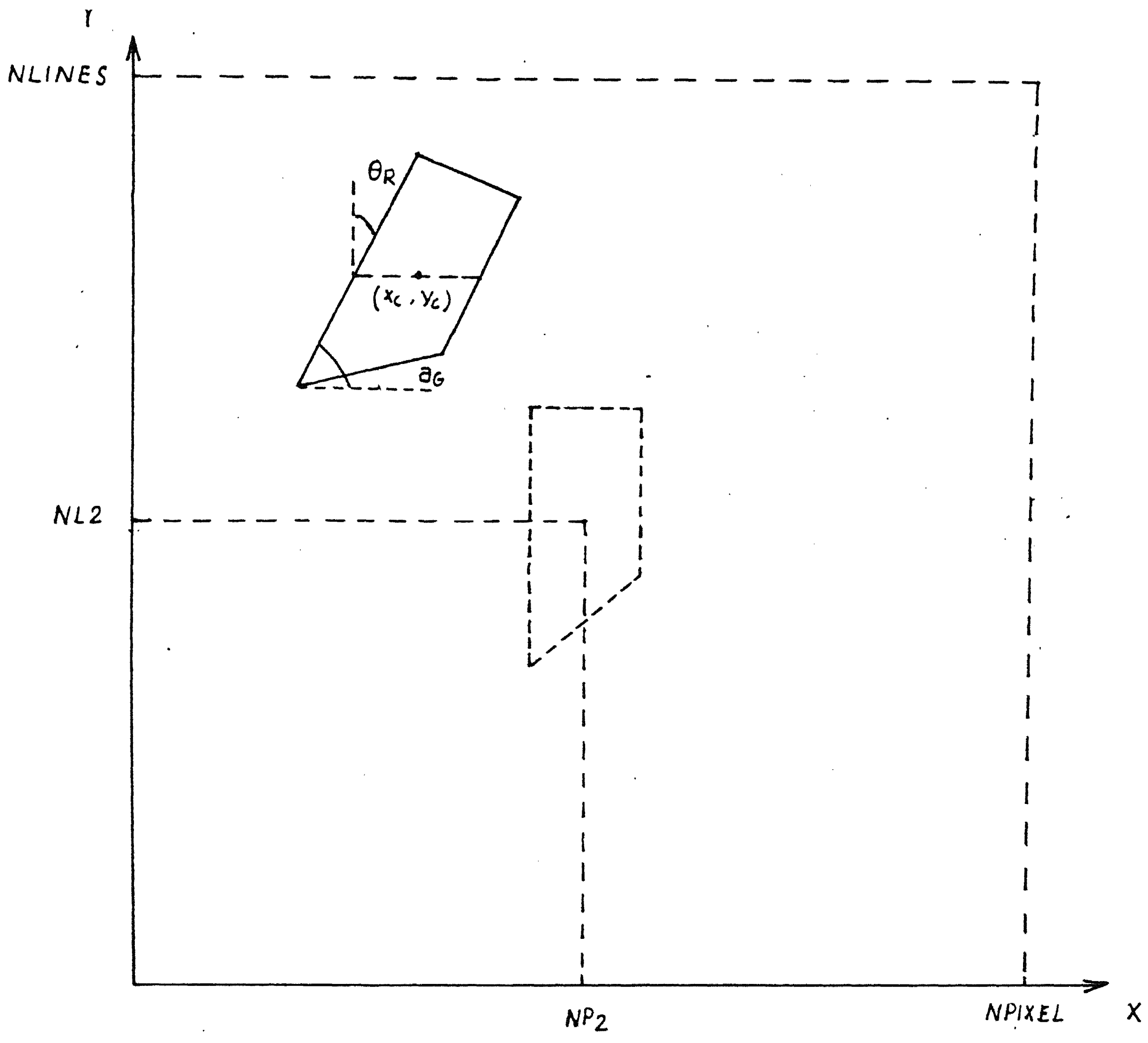

Fig. 4.3: Translation and rotation of the image

This simulates the positioning of the gripper.

The rotation of the gripper is also simulated. The deviation of the gripping segment from the y-axis is a measure of the rotation angle. The angle of rotation is given by

$$\theta_R = \pi/2 - a_G \qquad (4.4)$$

where,

$a_G$ is the absolute angle of one of the gripping segments.

The image is rotated anticlockwise by $\theta_R$ about the centre of field of view.

The transformed co-ordinates are given by the relation

$$x' = (x-NP2) \cos \theta_R - (y-NL2) \sin \theta_R$$
$$y' = (x-NP2) \sin \theta_R + (y-NL2) \cos \theta_R \qquad (4.5)$$

## 4.4 GENERAL PURPOSE DISPLAY CARD:

The circuit diagram of the display card is shown in Fig. 4.4. This display card has been developed jointly with a B.Tech. project done by Shri U. Guha.

It consists of:

i) a memory block for storing the pixel values,

ii) a set of counters to generate address for the memory and to obtain the sync. signals for the monitor.

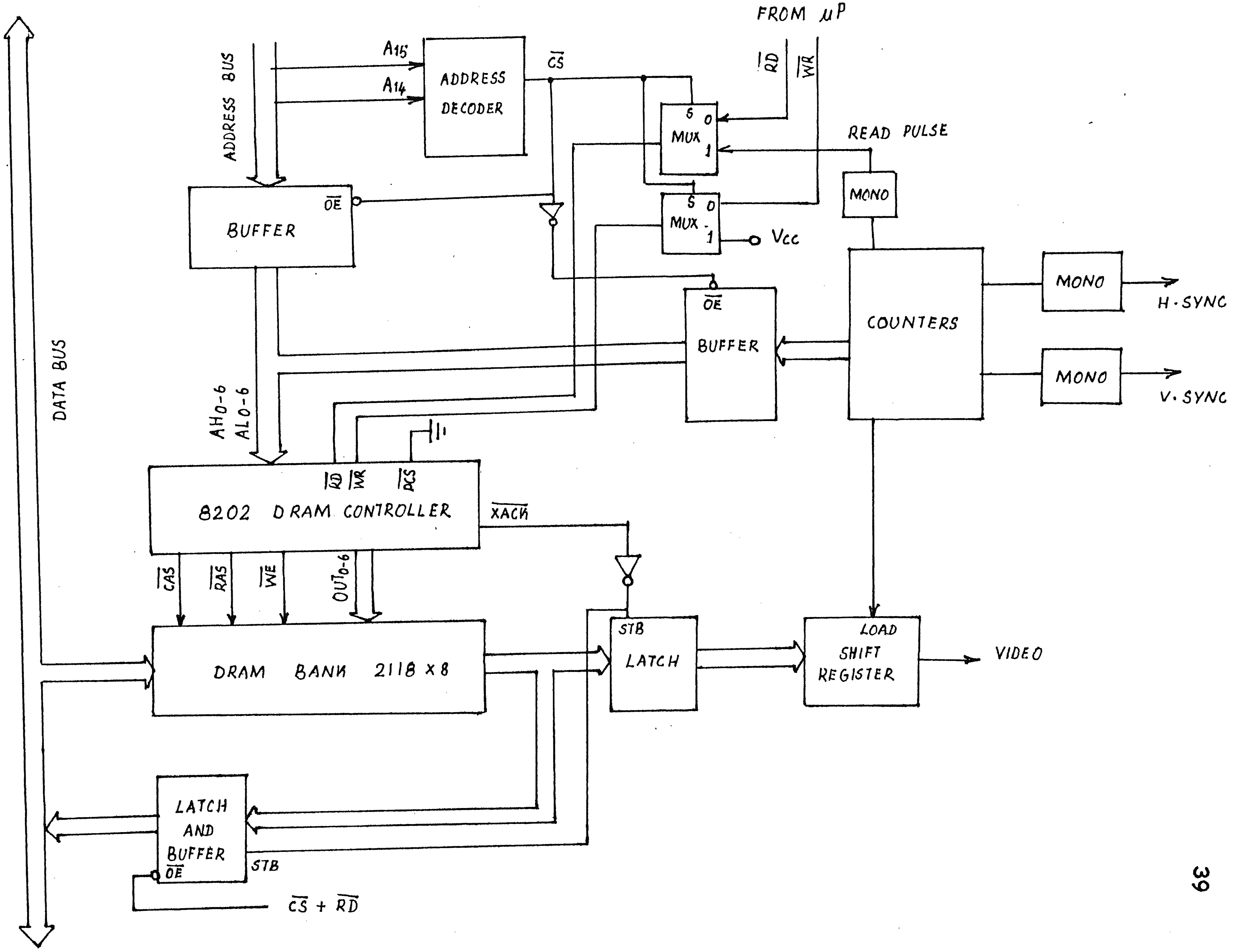

Fig. 4.4: Graphics display system

iii) Monostables for controlling the sync. pulse widths.

iv) A parallel-serial shift register for generating the video signal.

The horizontal scan time for a line is about 64 microseconds (Samsung Monitor specifications). With a blanking time of 12.8 microseconds, the visible line duration is 51.2 microseconds. So for a dot clock of 10 MHz, a horizontal resolution of 512 pixels is obtained. Providing one bit per pixel will result in a storage requirement of 64 bytes per line. So for the 256 active lines, the memory size required is 16K bytes. Due to the large memory requirement, dynamic memories are used to accommodate the whole circuit in one card of standard size.

The dot clock being 10 MHz, the time taken to shift one byte serially is 800 ns. When one byte is shifted out serially, the next byte in the pipeline is accessed from the DRAM (dynamic RAM) and stored in a buffer. Hence for every 800 ns a read cycle is performed.

The data for the display is generated by a software module 'DISPLAYDATA'. Using the vertices of the polygonal image to be displayed as the input, this module generates the byte-oriented data for the display. Due to the discrete nature of the display system, a staircase approximation is done to connect the points between two vertices.

# CHAPTER 5

# FLOWCHARTS OF THE SOFTWARE MODULES

The entire software is arranged in three main modules - learning process routine, recognition process routine and the robot control process routine. These routines are discussed in this chapter.

## 5.1 LEARNING PROCESS ROUTINE:

The flowchart of this routine is given in Fig. 5.1. The learning process involves the determination of the number of vertices and the co-ordinates of the vertices of the (polygonal) image of the object from the Video Data. The gripping segments of the objects are also specified by the user. For this purpose, the sides of the polygon are numbered clockwise starting from the bottom.

Having acquired the Video Data, the vertices of the polygon are determined. The other features like the sequence of relative angles and the sequence of lengths of the sides are also computed as explained in the next two sections.

It is possible for two different objects to have the same boundary. To prevent a possibility of error in the recognition process, new objects which can be mistaken for some other stored object are rejected during the learning stage. Otherwise, the features of the new object are stored.

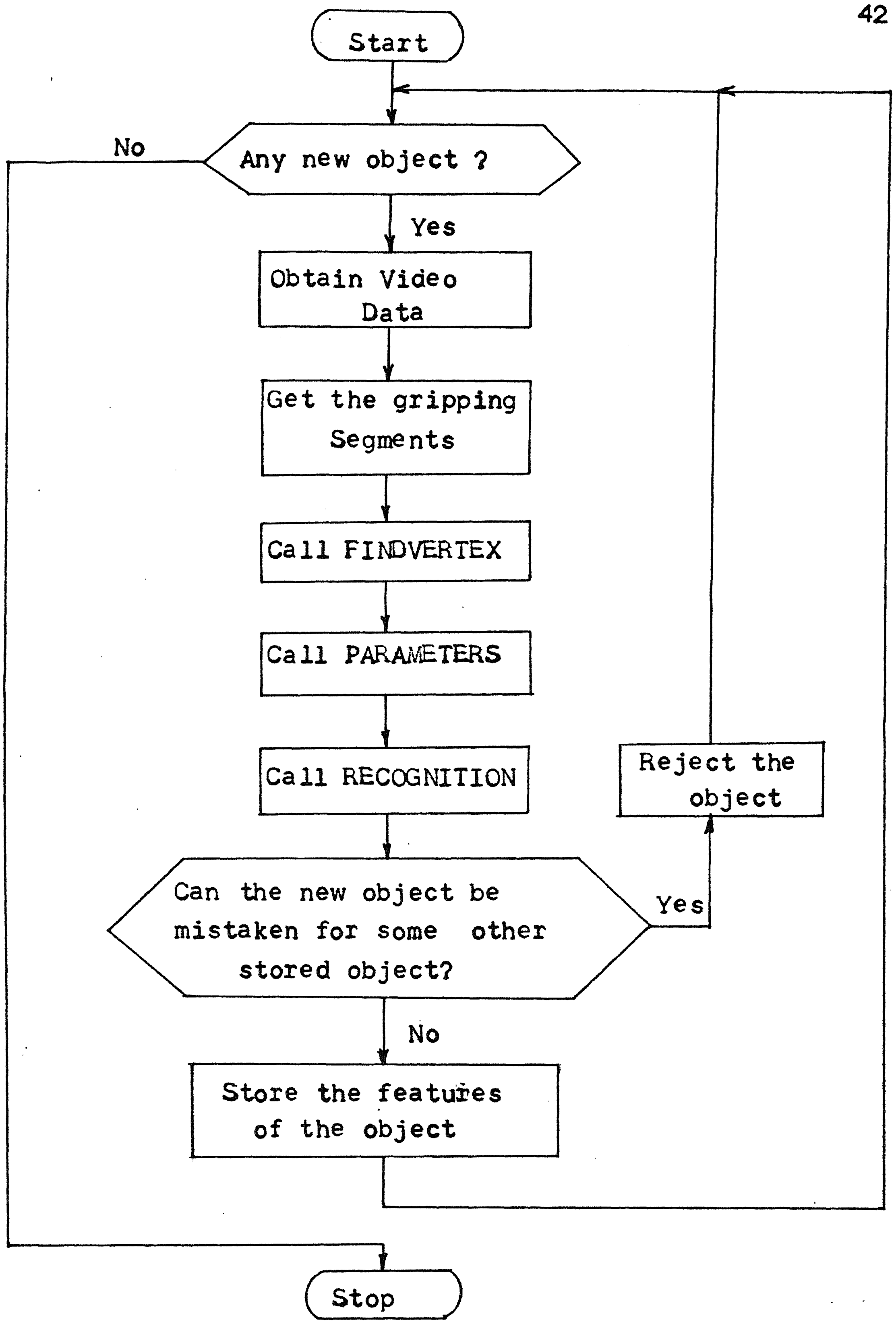
Start
Any new object ?
No
Yes
Obtain Video Data
Get the gripping Segments
Call FINDVERTEX
Call PARAMETERS
Call RECOGNITION
Reject the object
Can the new object be mistaken for some other stored object?
Yes
No
Store the features of the object
Stop

Fig. 5.1: Learning Process Routine

The output file contains the total numbera of objects and the features of all the objects: number of vertices, gripping sides, the sequence of relative angles and the sequence of the lengths of the sides, stored in that order as the features ofthe object.

## 5.1.1 Determination of Vertices (FINDVERTEX):

The contour is traced clockwise starting with the lowest point; in case there is a horizontal line at the bottom of the contour, the point at the left end of the segment is chosen as the starting point .

Fig. 5.2 gives the meaning of the variables used in the flowchart in Fig. 5.3. IY denotes the Y-co-ordinate of the lowest point of the contour and FY denotes the Y-co-ordinate of the highest point of the contour. NY denotes the Y-co-ordinate of any contour point. As the contour is traced upward, NY increases from IY to FY. This is denoted by UP=1. As the contour is traced downward, NY decreases from FY to IY. Contour points at IY and FY specify two of the vertices of the polygon. The value of ND corresponding to IY or FY denote the presence or absence of a line segment at IY or FY. If it is present, then both the vertices of the line segmat are known directly. The remaining vertices are found using the principle explained in Sec. 3.5.

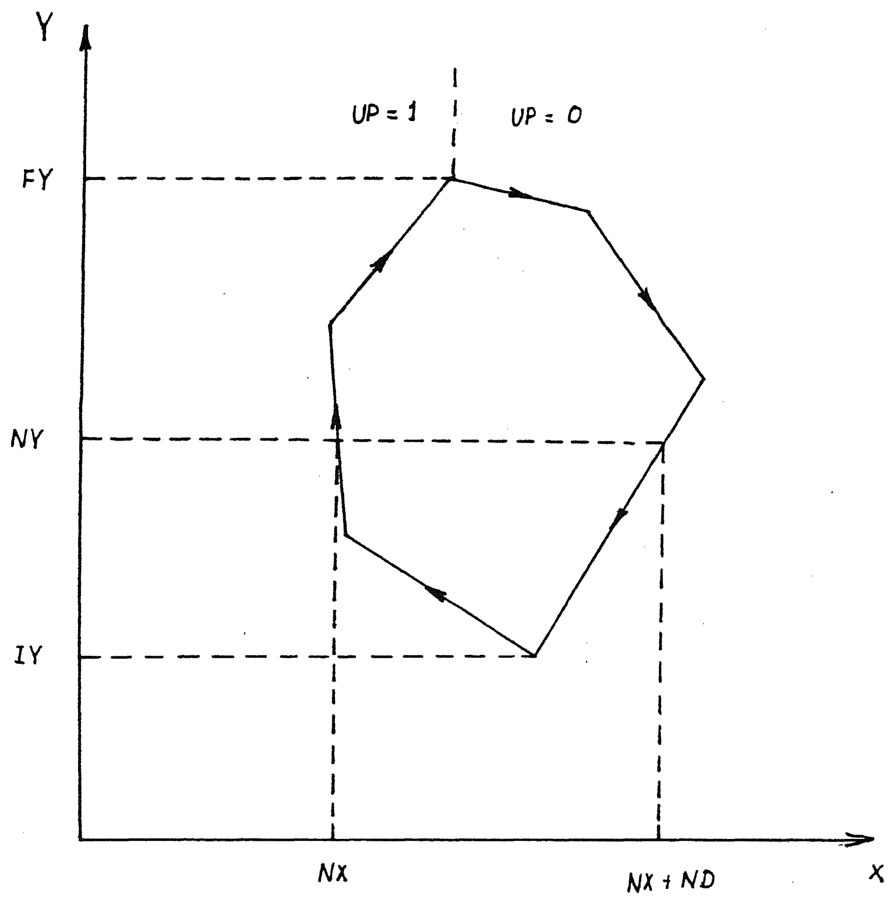

Fig. 5.2: Definition of variables used in FINDVERTEX

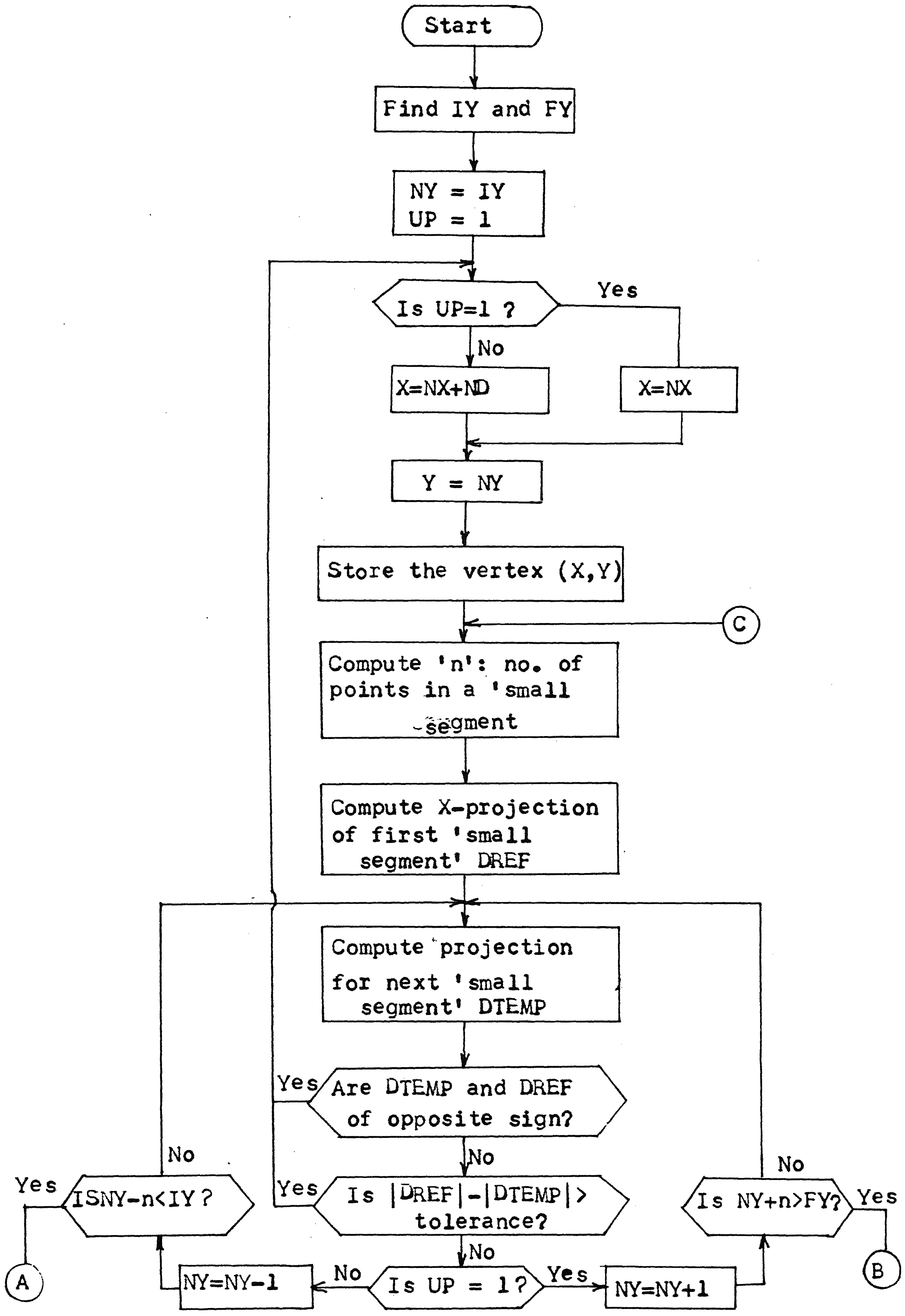
Start
Find IY and FY
NY = IY
UP = 1
Is UP=1 ?
Yes
No
X=NX+ND
X=NX
Y = NY
Store the vertex (X,Y)
C
Compute 'n': no. of points in a 'small segment
Compute X-projection of first 'small segment' DREF
Compute projection for next 'small segment' DTEMP
Yes
Are DTEMP and DREF of opposite sign?
No
No
Yes
ISNY-n<IY ?
Yes
Is |DREF|-|DTEMP|> tolerance?
No
No
Is NY+n>FY?
Yes
A
NY=NY-1
No
Is UP = 1?
Yes
NY=NY+1
B

Fig. 5.3 (contd...)

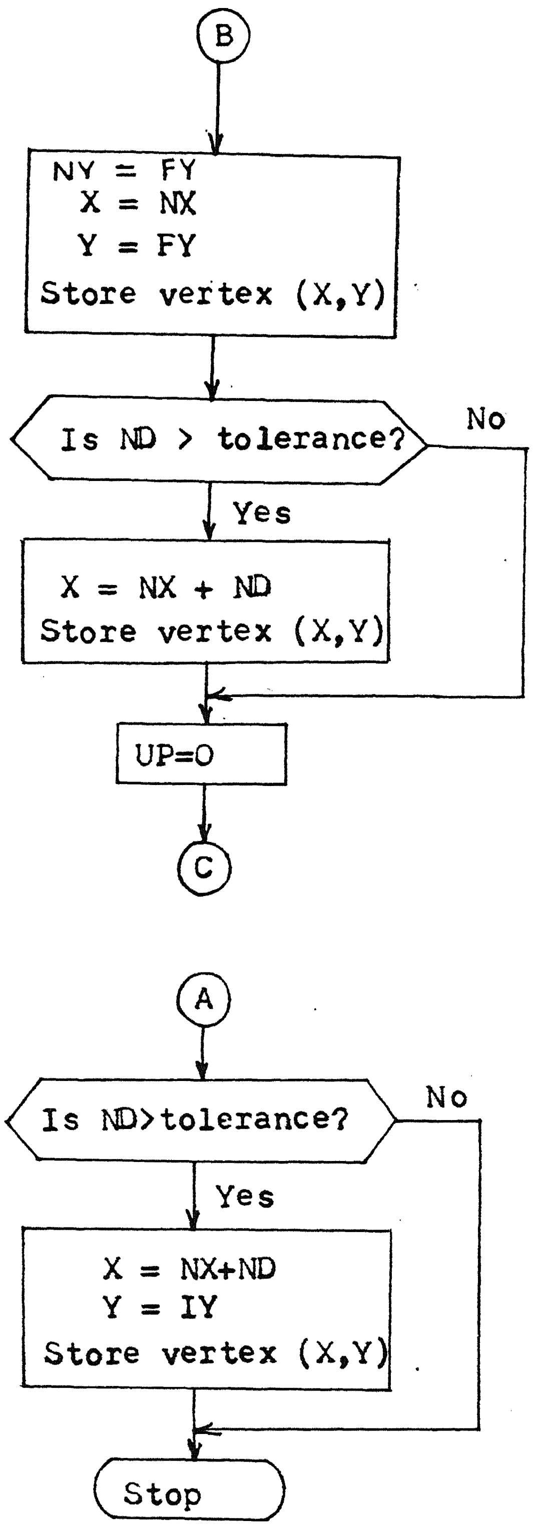

Fig. 5.3: Flowchart of FINDVERTEX

### 5.1.2 Determination of Features (PARAMETERS):

The computation of the features is fairly straightforward as shown in Sec. 3.5. The vertices of the polygonal image form the input.

The sequence of absolute angles, relative angles and segment lengths are computed by this module and fed to the main module. Fig. 5.4 gives the flowchart for this programme.

## 5.2 RECOGNITION PROCESS ROUTINE:

Fig. 5.5 gives the flowchart for this main routine. The modules FINDVERTEX and PARAMETERS explained in Sec. 5.1 are used to determine the features of the unknown object from the Video Data of the unknown object.

In the first step, it is checked whether objects with similar number of vertices are stored. If no object is found, then the unknown object is rejected and classified as 'unknown'. Otherwise the sequence of relative angles of all objects with similar number of vertices is matched with that of the unknown object. If no object with similar shape is found, the unknown object is rejected. If they match, then the segment lengths are compared. If they match, then the object is recognized.

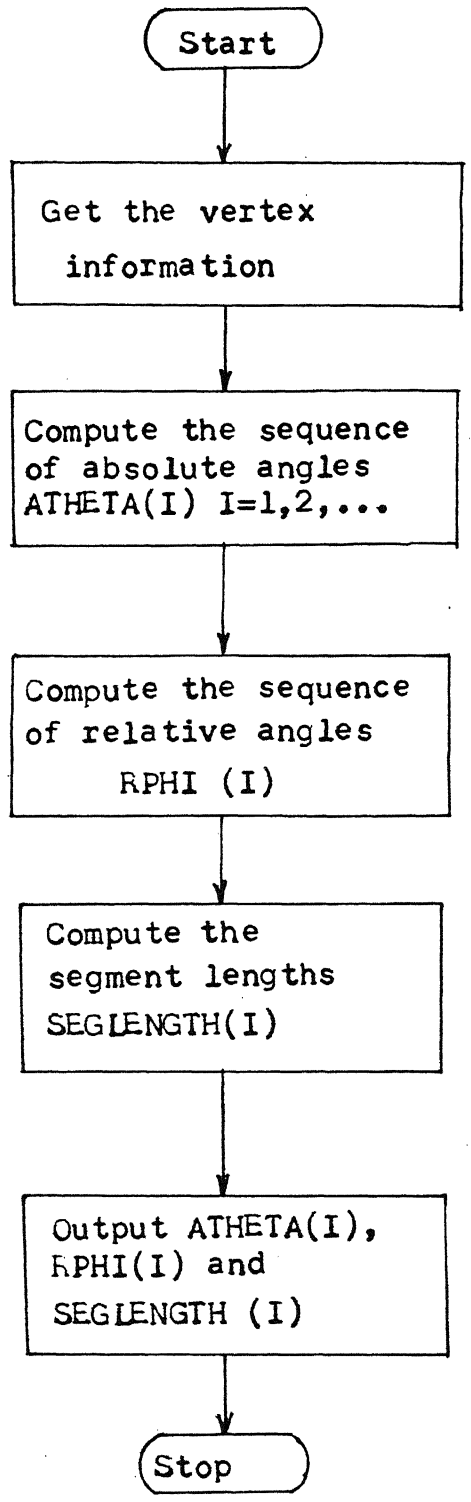
Start
Get the vertex information
Compute the sequence of absolute angles ATHETA(I) I=1,2,...
Compute the sequence of relative angles RPHI (I)
Compute the segment lengths SEGLENGTH(I)
Output ATHETA(I), RPHI(I) and SEGLENGTH (I)
Stop

Fig. 5.4: Flowchart of PARAMETERS

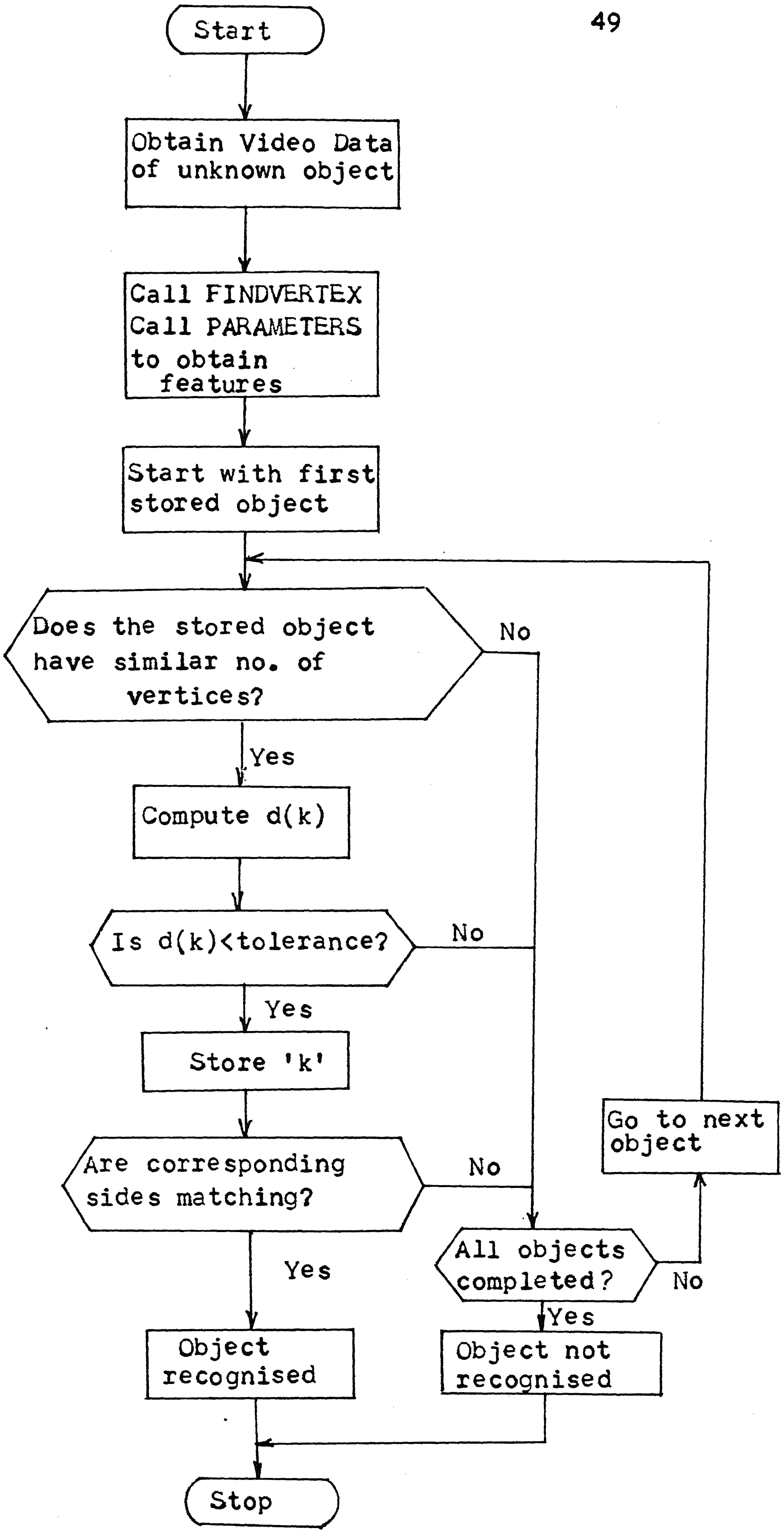

Fig. 5.5: Flowchart of recognition process.

As explained in Sec. 3.6 the shift factor 'k' is determined and fed to the next module. 'k' specifies the rotation of the unknown object with respect to the stored reference.

## 5.3 ROBOT CONTROL PROCESS ROUTINE:

The position and the rotation angle of the unknown object are found in this routine. Sec. 4.3.3 explains the simulation of the robot control process.

To determine the position of the object, the 'centre' of the object has to be computed. For this purpose, the gripping segments of the recognised object have to be found. If the gripping segments of the reference object and the new object are denoted by GR(J) and GU(J) respectively. Then

$$GU(J) = (GR(J) + k) \bmod NV \qquad \text{for } J = 1,2$$

NV denotes the number of vertices in the polygonal image of the object.

Using the information about the gripping segments, the 'centre' $(X_c, Y_c)$ is found. $T_x$ and $T_y$ (equation 4.2) give a measure of the movement of the gripper along X-axis and Y-axis respectively. But it will be of opposite sign. The gripper is rotated clockwise by an amount $\theta_R$ (equation 4.4). The movement along the Z-axis is controlled by the ratio of the sizes of the reference image and the new image. Fig. 5.6 gives the flowchart of this program.

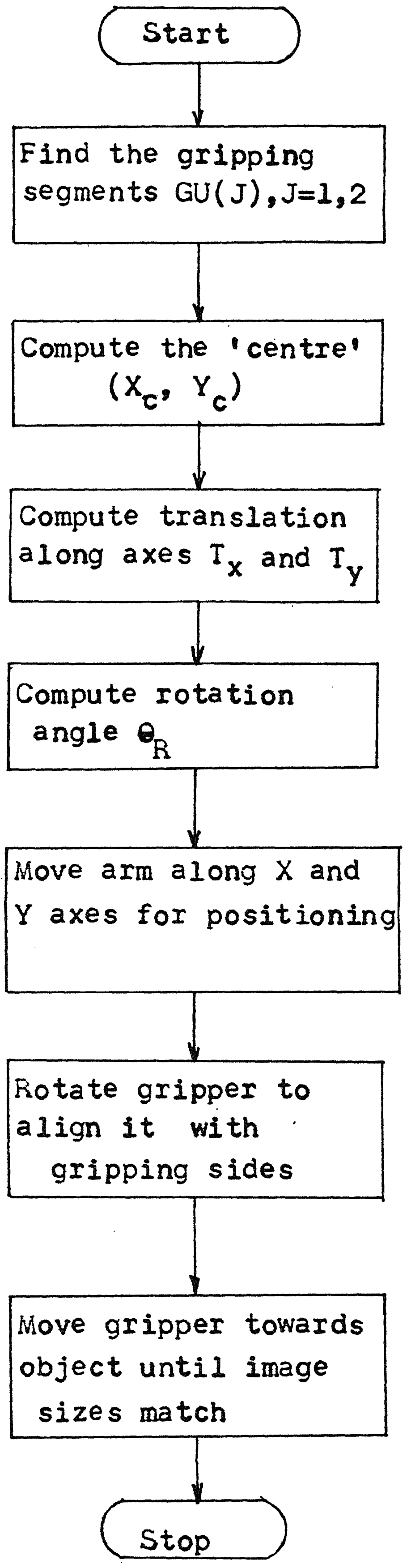
Start
Find the gripping segments GU(J),J=1,2
Compute the 'centre' $(X_c, Y_c)$
Compute translation along axes $T_x$ and $T_y$
Compute rotation angle $\theta_R$
Move arm along X and Y axes for positioning
Rotate gripper to align it with gripping sides
Move gripper towards object until image sizes match
Stop

Fig. 5.6: Robot Control Process Routine

## 5.4 GENERATION OF VIDEO DATA (OBTAINDATA):

In a real robot vision system, the camera acquires the image of the object. Video Data obtained from the acquisition module serves as the input to the learning and recognition routines.

In a simulated system, the Video Data has to be provided by the user. Due to the difficulty involved in feeding 2N words of Video Data, a software module has been written to generate the Video Data using the information about the vertices as the input. Instead of placing the object in its reference position and acquiring its image using a camera, the vertices of the object as in the reference position are specified as the input to the module.

The vertices are then re-arranged so that the lowest vertex is stored as the first and all other vertices follow in clockwise order. In case, there is a line segment at the bottom, the vertex to the left end of the segment is chosen to be the starting vertex.

From the vertices, the equation of each side of the polygon can be found. Given the Y-co-ordinate of a point on the line, the X-co-ordinate can be determined. It is then rounded off to the nearest grid point.

The X-co-ordinate of a point on the upward tracing of the contour (UP=1) gives the count NX for that line. The X-co-ordinate of a point on the downward tracing of the contour (UP=0) gives the value (NX+ND) for that scan line.

Let 's' denote the inverse of the slope of the line. The X-co-ordinate $x_i$ of the point during the upward tracing is given by the relation

$$x_{i+1} = x_i + s \quad \text{for} \quad i = 1,2,\ldots$$

where $x_1$ is the X-co-ordinate of a vertex. Since the contour is traced clockwise, the X-co-ordinate of the points during the downward tracing is given by

$$x_{i+1} = x_i - s \quad \text{for } i = 1,2,\ldots$$

The flowchart of the routine is given in Fig. 5.7.

For the recognition process, a rotated and scaled version of the reference image of an object is used as the input. The software module (ROTAT) is used to generate the Video Data of the unknown object. This module uses the previous module OBTAINDATA. But in this case, the polygon generated by the vertices is scaled and rotated about the 'centre'. The angle of rotation and the scaling factor are provided by the user.

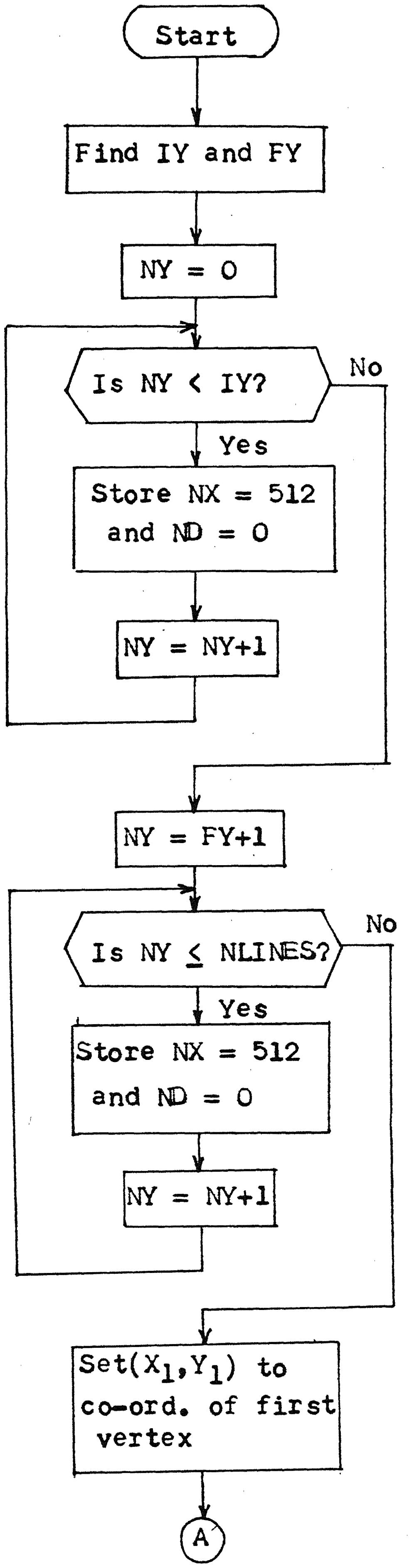
Start
Find IY and FY
NY = 0
Is NY < IY?
No
Yes
Store NX = 512
and ND = 0
NY = NY+1
NY = FY+1
Is NY ≤ NLINES?
No
Yes
Store NX = 512
and ND = 0
NY = NY+1
Set($X_1$,$Y_1$) to
co-ord. of first
vertex
A

Fig. 5.7: (contd...)

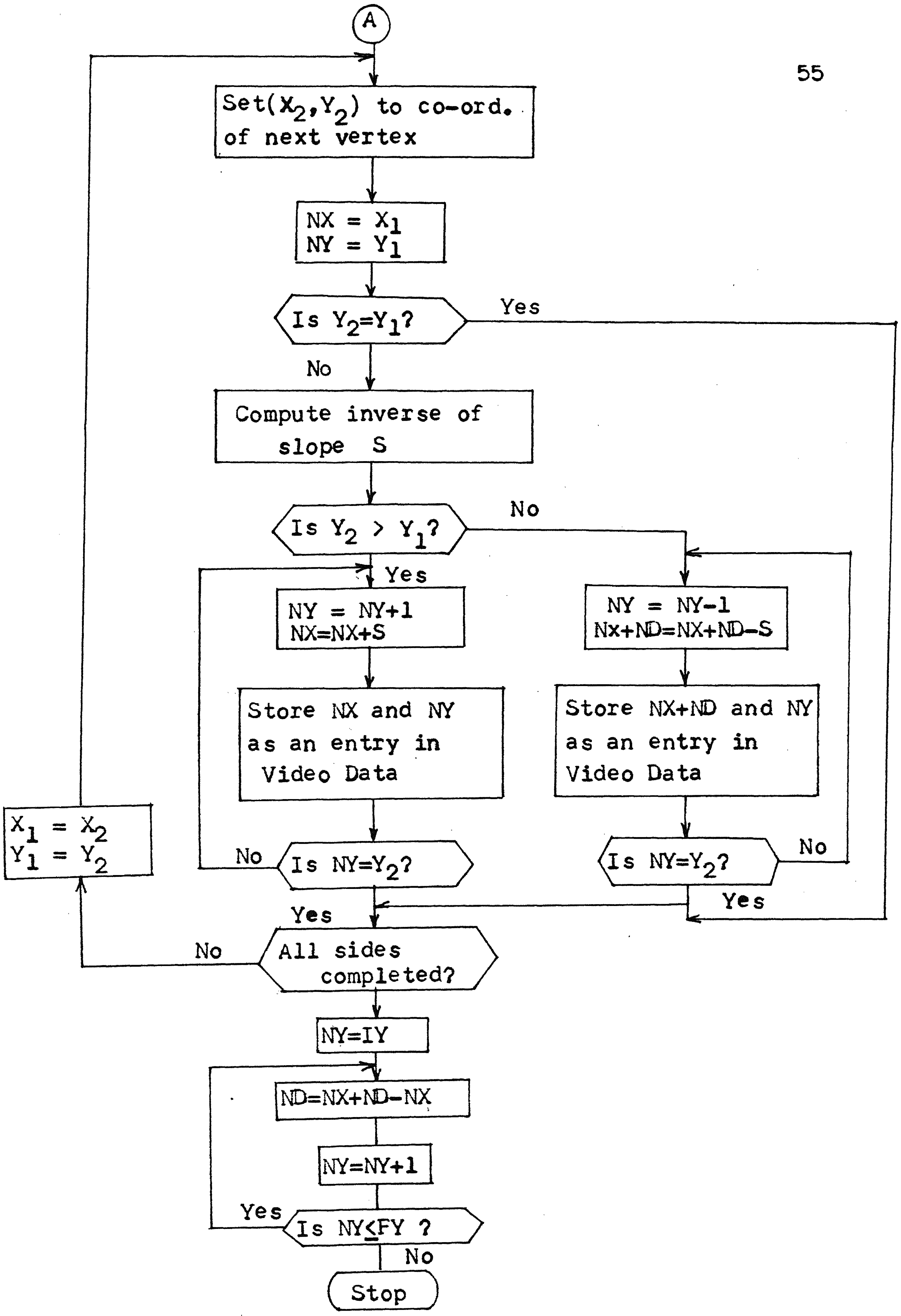

Fig. 5.7: Flowchart of OBTAINDATA

## 5.5 DISPLAY OF IMAGES ON THE VIDEO MONITOR (DISPLAYDATA):

The need for the display card has been explained in Sec. 4.2. The byte-oriented display data is used to generate the video image on the screen. In simulation, the acquired images of the objects are specified in terms of its vertices. So the information about the vertices can be used to generate the data for the display. The flowchart of the programme is given in Fig. 5.8.

Each side of the polygon is considered separately. For each incremental value in X the corresponding Y-co-ordinate is found and rounded off to the nearest integer. The cycle is repeated till all the sides of the polygon are completed.

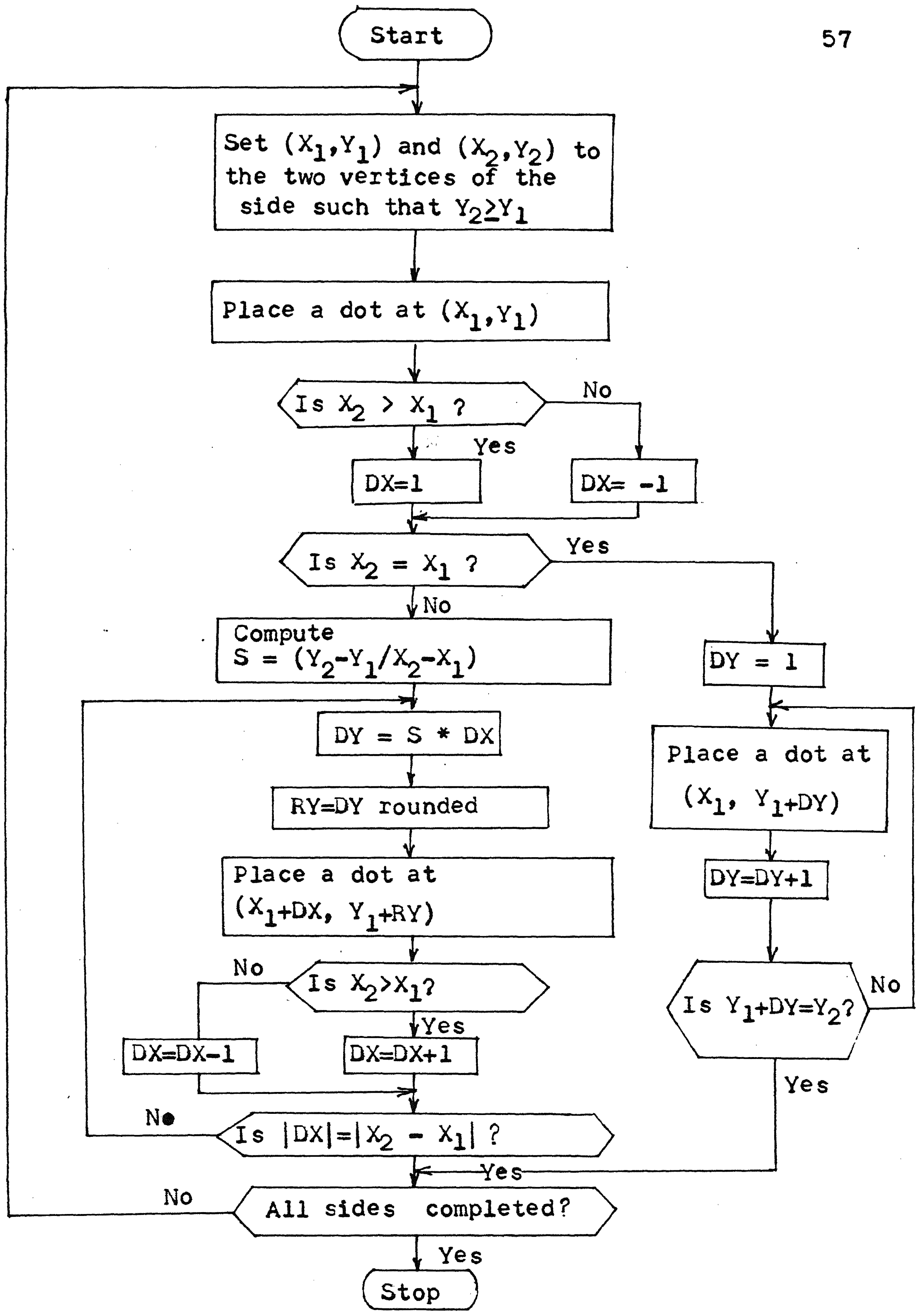
Start
Set $(X_1,Y_1)$ and $(X_2,Y_2)$ to the two vertices of the side such that $Y_2 \geq Y_1$
Place a dot at $(X_1,Y_1)$
Is $X_2 > X_1$ ?
No
Yes
DX=1
DX= -1
Is $X_2 = X_1$ ?
Yes
No
Compute $S = (Y_2-Y_1/X_2-X_1)$
DY = 1
DY = S * DX
Place a dot at $(X_1, Y_1+DY)$
RY=DY rounded
DY=DY+1
Place a dot at $(X_1+DX, Y_1+RY)$
No
Is $X_2>X_1$?
Is $Y_1+DY=Y_2$?
No
Yes
DX=DX-1
DX=DX+1
Yes
No
Is $|DX|=|X_2 - X_1|$ ?
Yes
No
All sides completed?
Yes
Stop

Fig. 5.8: Flowchart of DISPLAYDATA

# CHAPTER 6
# CONCLUSION

A robot vision system for object recognition and location determination has been developed.

## 6.1 RESULTS:

The two main modules - learning process and recognition process have been tested and are found to work satisfactorily.

A simple input-output sequence for the learning routine is given below:

```
Any new object ?  Y
No. of vertices : 4
Enter the vertices
100  25
 50  50
 45 100
200 100
Any new object? N.
```

The features of the object are stored in a separate file.

The sequence of operation for the recognition scheme is as follows:

```
No. of vertices of unknown object : 4
Enter the vertices:
100   25
 50   50
 45  100
200  100
Rotation angle?  = 135.0
Scaling factor?  = 0.8
Shift factor 'k' = 2
Object recognised.
```

The programme was tested for different angles of rotation and scaling factor of stored object and it recognised the object everytime.

## 6.2 SCOPE FOR FURTHER WORK:

The system can handle convex polygons having any number of vertices. The maximum number of vertices permitted can be modified by changing the parameter MAXVTX in the software modules.

The simulation of the robot control process has been done for the 'eye in hand' robot. But the acquisition and recognition modules can be utilised for situations where static overhead cameras are used.

The system can be extended to handle objects having arbitrary shape. The first natural step would be upgrade the software to handle any polygonal shape. A real system with a TV camera and the associated image acquisition hardware can be built and the software can be evaluated.

## REFERENCES

1. Alan Pugh, 'Robot Vision', IFS (Publications) Ltd., U.K., Springer Verlag, 1983, pp. 21-42.

2. Charles T. Zahn and Ralph Z. Roskies, 'Fourier Descriptors for plane closed curves', IEEE Transactions on Computers, vol. C-21, No. 3, March 1972, pp. 269-281.

3. Christer U. Peterson, 'An Integrated Robot Vision System for Industrial Use', Proceedings of the 3rd International Conference on Robot Vision and Sensory Controls, November,1983, Massachusetts, U.S.A., pp. 241-247.

4. W. Geisler, 'A vision system for shape and position recognition of industrial parts', Proceedings of the 2nd International Conference on Robot Vision and ensory Controls, Stuttgart, Germany, 1982, pp. 253-263.